Using Distributed Technology for Teaching Distributed Systems

David Willson Thornton B.Sc. CEng M.I.E.E.

A dissertation submitted to University of Dublin,

in partial fulfilment of the requirements for the degree of

Master of Science in Computer Science

Declaration

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:_____

David Willson Thornton

September 1999

Permission to lend and/or copy

I agree that Trinity College Library may lend or copy this dissertation upon request.

Signed:_____

David Willson Thornton

September 1999

Acknowledgements

I would like to thank my supervisors Mr Brendan Tangney and Dr Vinny Cahill for their enthusiasm, impulse and attention to detail over the course of this research.

Thanks also to my wife, Elaine, for her love and support.

Summary

Current developments in multimedia and Internet technologies are enabling their wide use as a means for delivering education and training. There is much diversity and no single architecture has yet emerged as the standard.

This dissertation presents a unified architecture for course development, design and delivery and a set of collaborative and multimedia tools to support it.

A course design and management architecture is discussed. A relational database lies at the centre of the system. Course developers use the tool via a standard database GUI; all students access the system via the Web. The course structure and presentation seen by the student is configured on a per person basis. The architecture was implemented and used to develop a demonstration course. The prototype demonstrates the application of technology to course design and administration, content development and collaborative activities. Its key features are openness, flexibility and a distributed architecture.

An evaluation of the prototype implementation showed that current technologies could be integrated to provide a unified architecture. A large number of commercial content development and conferencing tools are available to support educational activities. The set of common media formats supported by them allows a rich variety of content and interactive activities to be combined for presentation on a single client.

However, the range of existing multimedia formats and diversity in client applications must be addressed if the potential demonstrated by the prototype can be exploited fully and simply by all teachers. Work is on going in this regard and may be expected to produce results in the near future.

Table of Contents

Tables and Illustrative Material

List of Abbreviations

AI	Artificial Intelligence
ASF	Active Streaming Format
ASP	Microsoft's Active Server Pages
AVI	Audio Video Interleaved
AWS	Apache Web Server
CBT	Computer Based Training
CGI	Common Gateway Interface
COM	Common Object Model
CORBA	Common Object Resource Broker Architecture
CIF	Common Interchange Format
DCOM	Distributed Common Object Model
DHTML	Dynamic Hypertext Mark-up Language
DOS	Disk Operating System
GIF	Graphics Interchange Format
AHA	Adaptive Hypermedia Architecture
GUI	Graphical User Interface
HCI	Human compute Interface
HTML	Hypertext Mark-up Language
HTTP	Hypertext Transfer Protocol
IE	Microsoft Internet Explorer
IETF	Internet Engineering Task Force
IIS	Microsoft Internet Information Server
IMS	Instructional Management System
ITU	International Telecommunications Union
JPEG	Joint Picture Expert Group
JDBC	Java Data Base Connectivity
JiT	Just-in-Time
JMF	Java Media Framework
JWS	Java Web Server

MIME	Multipurpose Internet Mail Extension
MOV	Quick Time Movies
MPEG	Motion Picture Expert Group
MSBD	Media Stream Broadcast Distribution
ODBC	Open Data Base Connectivity
OHP	Open Hypermedia Protocol
IDL	Internet Definition Language
OLE	Object Linking and Embedding
OODBMS	Object Oriented Data Base Management System
ORDBMS	Object Relational Data Base Management System
OS	Operating System
PCM	Pulse Code Modulation
PNM	Progressive Networks Metafile
PDF	Portable Document Format
PPT	PowerPoint file format
RDBMS	Relational Data Base Management
RISC	Reduce Instruction Set Computer
RM	Real Systems Media format
RSVP	Resource Reservation Protocol
RTCP	Real Time Control Protocol
RTP	Real Time Protocol
RTSP	Real time Streaming Protocol
RMI	Java Remote Method Invocation Package
SDK	Software Development Kit
RAM	Real Audio Meta file format
RPM	Real Systems Plugin Metafile
SMIL	Synchronised Multimedia Integration Language
SMTP	Small Message Transfer Protocol
SGML	Standard Generalised Mark-up Language
SQL	Structured Query Language
SWF	ShockWave Flash file format
TCP	Transport Control Protocol

TCP/IP	Transport Control Protocol
UDP	User Datagram Protocol
URL	Universal Resource Locator
USB	Universal Serial Bus
WAV	Wave file format
WMF	Windows Meta File
XML	Extensible Mark-up Language
W3C	World Wide Web Consortium

References

[ADO99] *Adobe Acrobat 4.0, Promotional web page*, Adobe Systems Incorporated. URL http://www.adobe.com/prodindex/acrobat/main.html

[CAR97] *Enhancing Student Learning Through Hypermedia Courseware and Incorporation of Student Learning Style*, IEEE Transactions on Education (February 1999) pp33-38 [C. A. Carver, R. A. Howard, W. D. Lane]

[CHO99] *Developing Hypertext-Based Learning Courseware for Computer Networks: The Macro and Micro Stages*, IEEE Transactions on Education (February 1999) pp39-44, [C.Chou]

[CLA99] *Instructional System Design (ISD) handbook*, 1995 revised 1999 URL http://www.nwlink.com/~donclark/hrd/sat.html [D.Clark]

[CON99] *Educational Courseware Designer for the Web*. 4th Year Project, Department of Computer Science, Trinity College, Dublin, 1999 [O.Conlan]

[DIL98] *Courseware on the Fly*, M.Sc Thesis, Department of Computer Science, Trinity College, Dublin, 1998 [B.J. Dillon]

[DEB98] *Hypermedia Structures and Systems Hypertext and Hypermedia Course* 1998 version, Eindhoven University of Technology URL http://wwwis.win.tue.nl/2L690/ [P. De Bra]

[DEB99] *AHA! An Open Adaptive Hypermedia Architecture*. Eindhoven University of Technology [P. De Bra, L.Calvi]

[DEC98] *Towards a Generic Adaptive Hypermedia System*, Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia pp. 5-11, Ninth ACM Conference on Hypertext and Hypermedia, Hypertext '98, Pittsburgh, USA, 1998 [P. De Bra, L. Calvi]

[FAL99] *Home page*, Falkirk College, UK URL http://www.falkirkcollege.ac.uk/

[FEL] *Learning Styles and Strategies*. North Carolina State University. http://www2.ncsu.edu/unity/lockers/users/f/felder/public/Learning_Styles.html [R. Felder R. & B. Soloman]

[HOW96] *Felder's Learning Styles, Bloom's Taxonomy, and the Kolb Learning Cycle: Tying it all Together in the CS2 Course,* Paper, Proceedings of the twenty-seventh SIGCSE technical symposium on Computer Science Education 1996, ACM SIGCSE Bulletin Vol. 28, No. 1 (March 1996), Pages 227-231 [R.A. Howard, C.A. Carver, W.D. Lane]

[IET99] *RTP: A Transport Protocol for Real-Time Applications*, draft-ietf-avt-rtp-new-04.txt, Internet Engineering Task Force, Jun 1999

[INF98] *The Virtual Class Room, Info World Report* (Vol. 20, Issue 47, November 23, 1998) http://www.infoworld.com/cgi-bin/displayArchive.pl?/98/47/webtrana.dat.htm

[ISO96] *ISO Standard MPEG-1 Coding of moving pictures and audio,* ISO/IEC JTC1/SC29/ MPEG 98, September 1998, URL http://www.cselt.stet.it/mpeg/standards/mpeg-1/mpeg-1.htm

[LAW97] *The Web and Distance Learning: What is Appropriate and What is Not,* Report of the ACM/SIGCSE ITiCSE'97 Working Group on the Web and Distance Learning, Proceedings of the conference on Integrating technology into Computer Science Education , ACM SIGCSE Bulletin Vol. 29, No. 3 (Sept. 1997), Pages 144- [P.B Lawhead et al.]

[MIC99] *XML DOM Reference,* MSDN Online, Microsoft URL, 1999 http://msdn.microsoft.com/xml/reference/xmldom/start.asp

[MIL] *Reworking the OHP: the Road to OHP-Nav,* Proceedings of the 4th Workshop on Open Hypermedia Systems for the Ninth ACM Conference on Hypertext and Hypermedia, Hypertext '98. Aalborg University Esbjerg, DK: Dept. of Computer Science. (CS-98-01), June 1998; Uffe K. Wiil (eds) pp. 48-53; http://www.mmrg.ecs.soton.ac.uk/publications/archive/millard1998/html [D. Millard, S. Reich, H.Davis]

[MOS98] *Java Servlets* ISBN 0-07-913779-2 [K. Moss]

[MSD94] *The Component Object Model: A Technical Overview*, Microsoft Online Library, October 1994 [S. Willliams and C. Kindel]

[MSD97] *Deploying ActiveX Controls on the Web with the Internet Component*

Download, URL http://msdn.microsoft.com/library/

[NUR98] *An Agenda for Open Hypermedia Research*. Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia (Hypertext '98) Pittsburgh pp 198-206 [P.J. Nürnberg, J.J. Leggett, U.K. Wiil]

[OHS95] *OHSWG Compendium*, Open Hypermedia Systems Working Group, Nov 97 revise 1998, URL http://www.csdl.tamu.edu/ohs/ohswg.html

[SK89] *Hypertext Hands-On!: An Introduction to a New Way of Organizing and Accessing Information*, Addison Wesley, 1989. [B. Shneiderman, G. Kearsley]

[THO] *Distance Education over the Internet*, Proceedings of the conference on Integrating technology into computer science education, ACM ITiCSE '96 Pages 147-149. [P. Thomas , L.Carswell, M. Petre, B. Poniatowska, B. Price. and J. Emms]

[THO97 *Teaching over the Internet: the Future*. Journal Article, IEE Computing & Control (June 1997) pp. 126-142 [P. Thomas]

[W3C97] *HTML 3.2 Technical Reference Specification*, W3C, Jan 1997 URL http://www.w3.org/TR/REC-html32.html

[W3C98] *HTML 4.0 Technical Reference Specification*, W3C, Apr1998 URL http://www.w3.org/TR/REC-html

[W3C99] *World Wide Web Consortium Process Document* version 8 June 1999, Process Working Group, URL http://www.w3.org/Consortium/Process/

[W3X98*] Extensible Mark-up Language (XML) Technical Reference Specification 1.0*, W3C, version February-1998 URL http://www.w3.org/TR/REC-xml

[WAD] *Evaluating the Design and Delivery of WWW Based Educational Environments and Courseware*, Joint ACM 3rd Annual Conference on Integrated Technology into Computer Science Education (I.T.iCSE) and 6th Annual Conference on Teaching of Computing, published by ACM August 1998, pp243-248 [V.P. Wade, C. Power]

[WBT] *Home Page*, WBT Systems, URL http://www.wbtsystems.com/

Chapter 1 Introduction

There has been a rapid increase in the use of technology in support of training and education. This has been made possible by a dramatic fall in the cost of multimedia technologies and increase in the performance of computer networks. The Internet explosion testifies to the way in which business and people in general have taken to the electronic delivery of information. Questions are being asked as to whether the educational sector should adopt it with equal vigour.

1.1 Motivation

Business was quick to see Computer Based Training as a means to achieve savings in training budgets, while increasing the effectiveness and availability of course material. Multimedia was harnessed to provide student appeal and to offer new ways of presenting lessons. Several software manufacturers developed products to satisfy the need for easy-to-use courseware. Formats and standards were largely proprietary and different authoring metaphors emerged for creating courses. The Internet provides a unifying medium for delivering courses and courseware products are including tools to support the presentation of learning material using it.

Client-server technologies offered mechanisms for supervising training activities. This was in tune with the prevailing management style and training applications were integrated with other Human Resources tools. Technology was taking the lead.

But skills-based training and education are not the same thing and there was concern that some CBT course did not sufficiently conform to sound pedagogic methods. IT support for education should first and foremost satisfy clear educational objectives. However, it should also support institutional needs. There is a requirement for a unified educational courseware architecture .

The Internet and computer networks offer the prospect of education on demand delivered direct to remote students, any time any place. The concept of remote education is not new, however; developments in post, print and other mass media have permitted distance learning over the course of more than a century. The Open University (OU), for example currently offers courses to more than 150,000 students per year, with some courses delivered to up to 8,000 students with up to 40,000 assignments processed per week, [THO]. Yet, at the same time the number of students participating in full tertiary

education has not fallen. The OU has integrated its application of technology extensively in support of all its activities.

Technology is not only appropriate to distance learning; developments in multimedia technologies and network based collaborative and conference tools are also finding their way into the class room. Microsoft Office and its ubiquitous PowerPoint application have given teachers the means for producing high quality overhead slides for about a decade These are now familiar lecturer tools and, arguably, have not adversely effected the class room. But, new tools are emerging rapidly and it is difficult to know which will be the most effective and simple to use. An evaluation of these tools is necessary.

The student should be centre stage in an educational environment. He should be able to participate fully in all learning activities. The institution must endeavour that its courses and lessons are accessible to all students. The lecturer should respond to the needs of a student and adapt the presentation format to suit. Thus openness and flexibility are important characteristics of teaching. The Internet satisfies both of these criteria but a closer examination of it is necessary before embracing it as the way forward.

A fresh look at education in the information age is required. What remains central to education is the pedagogic approach but technology should be employed to support it. A review of the many different technologies is required with a view to identifying those most suitable for education. From that innovative ways for apply them to learning may emerge and architectures to support them.

The aim of this dissertation is to examine the state of the art of distributed technologies, in particular those related to the Internet, as can be applied to the world of education and training. Pedagogical requirements will be established and opportunities to exploit appropriate technologies will be identified. It will be shown that current technologies can be combined in imaginative ways and with relative ease to satisfy specific learning objectives. Furthermore, it will be shown that technology can assist with the process of defining those objectives and supporting the provision of education.

A unified distributed architecture will be presented comprising components capable of cross platform required to support the system and open standards to ensure a consistent level of student access.

1.2 Holistic Approach

Much work and study has already been conducted into how specific technologies can be applied to training and education [THO]. This dissertation does not intend to reinvent the wheel. A holistic approach will be taken. The results of previous research [DIL98] into learning theory will be presented and implications taken as to how education should be structured and lessons conducted. Distributed technologies applicable to education will be described and assessed individually. An architecture will be developed and implemented using commercially available applications.

A prototype course will be created. It will be developed from conception to delivery against a backdrop of pedagogic and institutional requirements. The course will be delivered via a TCP/IP network and will demonstrate the integrated use of multimedia and collaborative tools.

An technical evaluation [WAD98] of the system will be conducted with a view to identifying areas for further research.

1.3 Pedagogy

There are three main learning theories: behaviourism, cognitivism and constructivism [DIL98]. These emerged chronologically but all still contribute to how education is provided.

Behaviourists argue that animals and humans can be conditioned and respond predictably to stimuli. Conditioning implies that repetition reinforces learning and requires activity. Cognitivism suggests that learning is based more on knowledge than experience and that knowledge conforms to the structure of the mind. Thus information is processed by the individual and well structure information will be learned most easily. Constructivists extend this and suggest not only that a mental structures exist, but that this structure is dynamic and adapts as individuals learn. Information is processed in a very personal way. Thus the presentation of information must be adapted and structured to the needs of the individual if it is to be learned most easily.

It can be concluded from the above that material should be structured, that its presentation be adapted and that learning activities should support reinforcement. Felder [HOW96] presented a structure for tailoring learning activities to suit preferred learning styles.

Bloom's analysis provided a framework for setting individual and group learning objectives.

1.4 Technology

Technology will considered under three headings: information content creation, content delivery and information organisation.

Multimedia includes a wide variety of information formats. At the most elemental level it is text on paper, but the range of possibilities extends through many graphical, audio and video formats. Conferencing permits audio, video and even desktop applications to be shared in real time. Common to them all is a requirement for capture and encoding. This requires specific tools and several of these will be evaluated. Web technology will be described as a means for delivering multimedia content and issues associated with presenting HTML on heterogeneous browser applications will be discussed. Information may be organised and structured in several ways. The Internet provides a very loose information structure. Relational databases support tight hierarchical structures and permit information to be stored and retrieved very efficiently. However their configuration should be planned. The application of database technology for providing a efficient access to information on the Web will be discussed and an implementation method offered.

1.5 A Unified Distributed Architecture

A unified distributed architecture will be presented. Course structures and administrative information will be organised in a relational database; Web servers will store and deliver static learning material; media servers will stream live and archived audio and video; conference servers will facilitate collaborative audio, video and data activities; the Internet will act as the binding agent and students will access the educational environment using a web browser.

1.6 Document Outline

This chapter has set the scene for the dissertation. It outlined the need for further research into IT support for education and outlined an objective for the research. Some of the issues relating to pedagogy and the management of multimedia information on the Internet were presented in brief. The remaining chapters will be describe the following:

Chapter 2. A summary of multimedia content formats and conferencing technologies set against their delivery over the Internet; database management systems and how Web access can be provided to them; the role that HTML plays in providing a common container for delivering information and data over the Internet and some complexity issues concerning the inconsistent presentation of content using different of web clients.

Chapter 3. A summary of the main learning theories; an introduction to the significance of Bloom and Felder in setting learning objectives and modifying presentation styles and the requirement for both individual and group learning activities.

Chapter 4. How some of the technologies described in chapter 2 can be utilised to meet the pedagogic requirements of chapter 3; the emergence of an information structure which is best supported by relational database technology

Chapter 5. A critique of several commercial courseware applications, multimedia authoring tools and online teaching material created using them; a summary of the research presented in the preceding chapters and a statement of requirement for an implementation architecture.

Chapter 6. An outline of the unified system architecture including details of database structure and access, capturing and handling streamed video, creating animations, authoring HTML and configuring web pages to create an integrated learning environment.

Chapter 7. Using the model to design and deliver a prototype online course within an institutional context; tools support for defining and developing a course structure; creating learning activities; student registration, class management and support for user profiles.

Chapter 8. An evaluation of the model in terms of its openness and the flexibility it provides to academics, administrators and students; an assessment of the consequences of using a distributed architecture; unresolved difficulties and recommendations for further research.

Chapter 2 Technology

2.1 Introduction

In this chapter we will examine some of the technologies which can be exploited to support the creation and delivery of learning material. This will be set in a context of delivery over a TCP/IP network.

The examination will begin with the Internet and HTML its de facto presentation standard. Traditionally, information has been presented textually and stored statically in a library. Different electronic formats will be discussed each adding specific value to basic text documents. Some of the many multimedia technologies currently available will be considered in terms of the benefits they offer and their associated costs and benefits. Support for the social aspects of an educational institution will be addressed by examining a range of collaborative and conferencing tools. Finally we will look at infrastructural issues concerning how content is delivered. Attention will be paid to host platforms, clients, servers and middleware applications.

2.2 The Internet and the World Wide Web

The popular image of the Internet belies its sophistication and potential. It is a true distributed system, a network of heterogeneous and independent computing machines, with equally heterogeneous communications links spanning the network. An educational resource which uses the internet as its primary means of delivery must recognise the opportunities and limitations which this diversity presents and be selective and imaginative in its exploitation.

2.3 The Hypertext Mark-up Language (HTML)

HTML provides a mechanism for sharing documents and navigating between documents over a network. Its origins lie in the academic and research domain. It uses mark-up within a plain text document to provide content formatting and document linking. Shneiderman [SK89] defines hypertext as "a database that has active cross-references and allows the reader to "jump" to other parts of the database as desired". It is made up of documents comprising nodes, the information content, and hyperlinks, the mechanism for moving between documents. Hypertext structure is very different to that of more

conventional query-type databases. For simplicity from here on, the term database will be used to describe the latter.

The use of mark-up was not new. Proprietary software applications already used it, but HTML offered a specific form of the Standard Generalised Mark-up Language (SGML) which could be considered an open standard and could be presented as text using a standard client or browser.

HTML developed by consent, with control over its standards exercised by a non-commercial body, the World Wide Web Consortium (W3C). The standards development process is outlined in its Process Document [W3C99]. The open nature of HTML made it an attractive text delivery vehicle for distributed networks. However, the influence of browser manufacturers challenged the open standards nature of HTML and led to proprietary implementations. This change is important and makes implementing universal HTML delivery quite complex.

2.3.1 HTML Development.

HTML standards represented the agreed state of the art in terms of the document format and of how web clients may be expected to present it. The nature of W3C, in particular it search for consent within the development community led to the standard lagging behind the cutting edge of delivery technologies. The emergence of leading and competing players, in particular Microsoft and Netscape, led to difficulties in maintaining the consent necessary for having an open standard. Client capabilities quickly surpassed the standard. This has resulted in competing proprietary implementations with their associated script languages.

2.3.2 HTML 3.2

HTML [HTM97] uses tags to provide document formatting and linking. Hyperlink tags include a reference to the next HTML node (HREH="xyz.html") which can be retrieved using the Hypertext Transfer Protocol (HTTP) an application layer protocol of the TCP/IP stack.

Formatting was initially limited to support for text, however HTML 3.2 provided for greater control over fonts, the inclusion of tables, text flow around images and the use of hot spots to link selectively from within images. Furthermore it introduced the ability to

embed Java applets (platform independent client side applications which will be discussed later).

The availability of low cost multimedia content gave the public a thirst for accessing it over the Internet. Browser technology was enhanced to support it. But the explosion in the diversity of multimedia content and problems encountered in presenting it on the client led to browser specific implementations. As a result, although an HTML 3.2 conforming browser could be assumed to give a fairly consistent presentation of text and still image and support for Java applets, additional support in particular with regard to audio visual media could not be assumed.

2.3.3 HTML4.0, Scripting and Dynamic HTML

So far, HTML has been a vehicle for presenting content. The manner in which content is displayed is consistent within the configuration of the browser installation only. HTML4.0 [W3C98] introduced Cascading Style Sheets as a mechanism for achieving presentation consistency across browsers. Client interaction was restricted to navigating a to subsequent node or sending client information to a server from an HTML form. Other types of client interaction were not possible. Scripting offered a solution and was incorporated into HTML4.0.

Browser developers gave web clients the ability not only to extract presentation information from HTML but also behaviour. This behaviour could be defined using a scripting language, which the browser could parse as the page was loaded and interpret in terms of document behaviour. But the interpretation was browser specific and led to the emergence of incompatible JavaScript and VBScript.

The existence of different scripting languages for specific browsers has a profound impact on how HTML content is created. If universal delivery is required, then either a lowest common denominator should be used, or intelligent scripting should be incorporated to identify the browser being used and to adapt the content to suit. Current scripting languages provide such mechanisms to assist the content creator.

Scripting is an important , however a detailed analysis is beyond the scope of this dissertation.. What is key to this study is that VBScript is supported by Microsoft and its client Internet Explorer and that JavaScript is primarily associated with Netscape and its Navigator client.

The concept of behaviour has an association with object technology. This association is reinforced by developments in XML. XML provides a structural framework to a document. As with scripting, the 2 main browser camps have their own interpretation. Microsoft uses the Document Object Model [MIC99] to identify document components each of which can he handled separately. Netscape uses layers. With a combination of layers and scripting, HTML pages can display dynamic properties, for example HTML animation. Such effects are representative of Dynamic HTML (DHTML)

Thus HTML has developed beyond the state of being just a presentation mark-up language to one which can define a document in object oriented terms. A document can have behaviour and can contain other object each with its own behaviour. Behaviour is implemented at the client using scripting. Scripting does not conform to an open standard and this creates problems for HTML authors. Objects within documents may be other documents; they may also be handler applications for presenting, for example, multimedia content.

2.4 Multimedia Information Formats

Multimedia is now used extensively by computer applications, be it static text, audio, graphic or movie content, dynamic streamed audio or video or a component within a shared conferencing environment. It may be delivered in its raw state or compressed. And the latter may use lossy or lossless compression-decompression (codec) algorithms. Lossy algorithms decompress to produce an approximation to the original media content - the higher the compression the worse the approximation. The palette available to the content author is vast, so a consideration of the available formats is necessary.

2.4.1 Textual Content

Hypertext.

The basic delivery medium for information is text. We have already seen that text can be formatted using HTML tags and displayed using a browser. Images, pictures and diagrams, can be embedded in the page and links can be included to permit navigation between documents. The final presentation of the document is subject to browser and screen display configuration. Cascading Style Sheets provide greater control over the use of fonts. The total document size is the sum of its parts, which can be quite large if

images are included. For users with slow dial-up connections, hypertext may not be the preferred format.

Portable Document Format.

Portable Document Format (PDF) files are frequently used for the delivery of text documents. A document is created using standard text editing tools and images can be included. A tool is required to convert it to *pdf* and also to view it at the client. Adobe Acrobat [ADO99] is available for the main operating systems. The appearance of the whole document is consistent across all platforms, it is contained within a single file and it can be saved or printed easily.

Proprietary Text Formats.

Proprietary format text files, for example Microsoft Word, can be delivered over the Internet. The application or a special viewer is required to process the document. Open University assignments are processed using Word 95 (or later versions). Using the track changes utility [THO97] tutors' corrections can be added without changing the original text. Proprietary format documents tend to include more mark-up than plain text, *pdf* or other formats and consequently tend to be much larger. They are not suitable for large-scale distribution.

2.4.2 Graphics Formats

Bit Map Graphics

Bitmaps represent images in the same way as does a computer display, with data representing each dot or pixel. The amount of data stored is dependent on the colour depth. This determines the number of possible colours renderings: 4-bit (16 colours), 8-bit (256), 16-bit (64,000), 24 bit(etc).. A quarter screen image for a 640x480 pixel display would require 75kbits of data per bit of colour depth. Consider downloading a file using a 56Kbps dial up modem; with the operating a full speed a 16-bit depth quarter screen image would be delivered in 22 seconds. One way to minimise file size is to reduce the graphic size. But, a quarter size image on a 640x480 display would be less than tenth-size on a 1024x768 display. Client software can be used to resize the image, but resizing leads to a loss of quality.

Vector Graphics.

Vector graphics store representations of images. The most common format is the Windows Meta File (WMF) used to create clip art. Because metadata is stored, file sizes tend to be much smaller that for bitmaps. Client software generates the image on demand. A further advantage is that images can be resized with ease and without any loss to image quality. Vector graphics are well suited to the production of diagrams.

Compressed Graphics.

Vector graphics cannot be used for photographs or other analogue images. Thus compression is required to reduce the file size.

Graphics Interchange Format (GIF). GIF is a lossless codec for images. However it only renders 8-bit colour, but supports transparency in its presentation. It uses run-length encoding and is thus more efficient for flat graphics than to messy real-life images. Colours beyond its 256-colour palette will be approximated.

JPEG Images. The JPEG format, *jpg*, codec can produce 24-bit image files smaller than *gif* files. This can be done because the JPEG codec is lossy. Because photographic quality is rarely required for screen presentation, especially if the frame size is reduced, *jpg* is used widely for representing photographs.

2.4.3 Audio Formats

Uncompressed Audio.

Wave files, *wav*, are digital representations of real analogue sound and are created by sampling a sound source, such as a microphone. The graduation given in bits defines how accurately the amplitude is measured and is analogous to film resolution or colour depth. Compact disk quality requires a sample rate of 44.1Khertz and 16-bit graduation. Raw sound generates large files: 16-bit CD quality stereo sound requires 172Kbits per second. This is inappropriate for real-time delivery over a modem. It could also lead to congestion on a LAN. Even telephone quality sound (mono 8-bit depth at 8 kHz sampling) requires 8Kbits per second. High quality recording can be made in *wav* format and played on computers with sound cards installed. However, they should be compressed if they are to be delivered over a network

Compressed Audio.

There are many audio codec systems available. The Microsoft sound recorder found on the Windows 95 interface can take advantage of several. A comparison of *mpeg*3 to *wav* (PCM) appears at table 1. Other codecs can provide even higher compression ratios. But, computers must have the appropriate codec software installed to play compressed sound. Thus consideration of the end user is necessary.

Format	Compressed	Quality	Sampling (Hz)	Data rate (bits per second)
PCM	No	16-bit stereo	24k	94k
MPEG3	Lossy	16-bit stereo	24k	5k

Figure 2.0-1Comparison of audio file sizes for compressed and uncompressed media

2.4.4 Static Video Formats

Video data is subject to the same data restrictions as is audio. In its raw format video is a series of bitmaps. Movie formats include sound as well.

Audio Video Interleaved (AVI).

AVI files, a Windows format, are sequenced bitmaps with *wav* files interleaved. They are easily created and rendered, but are large and impact badly on network traffic. An avi file recorded with a 160x120-pixel frame and 16-bit colour captured at 30 frames per second with interleaved CD quality sound would consume over 9Mbits per second. This can be reduced by cutting the colour depth, frame rate and frame size, but with a commensurate cut in quality

Quick Time Movies (MOV).

Quick time is the Apple equivalent of the avi format. It is well established and can be played on Windows and MAC platforms. It suffers from the same limitations.

Motion Picture Expert Group (MPEG).

MPEG employs a lossy codec for audio and video content to support data rates of up to 1.5Mbps.. It is an open standard; MPEG1 [ISO96] is the most popular, although there are currently 4 versions available. It takes advantage of the similarity one frame has with its

direct successor and of the predictability of that change and uses. An I-frame (Intra-frame, comparable to a *jpg* image with the same compression properties) is recorded followed by P-frames and B-frames (representing what has and what will change). The degree of compression achieved is based upon the final frame size (in pixels) the frame rate and ratio of I, P and B-frames. MPEG movies be encoded directly from live source or video tape (in-line encoding) or by converting from an *avi* file. It is not possible to specify the compression ratio universally, however a 21.5 second animation *avi* clip (328x236-frame) was converted to *mpg* (160x120). comparative file sizes were 9,530 and 238kbytes respectively. Sound can be compressed with or without a video content. Thus MPEG formats also exist for audio.

2.4.5 Streamed Multimedia Formats

Even compressed video files can take a long time to download before they can be played. Streaming permits media to be played while it is being downloaded. Streamed multimedia is buffered by the host, played and finally discarded. Streamed media can be produced live (synchronous) or served from a file (asynchronous). Live content is not truly synchronous. The act of buffering puts a delay, or latency, into the delivery process. Its purpose is to ensure the client has sufficient data to begin playing the content smoothly. Latency is typically 15 seconds for live content. Encoding can be tailored to suit network bandwidth conditions.

Streamed multimedia files require special players to deliver the content and to manage the flow of data between server and player. Data transfer can be achieved using HTTP, TCP and UDP. The Internet Engineering Task Force (IETF) developed RTP (Real Time Protocol), a time stamped derivative of UDP with sequence numbering, RTCP (Real Time Control Protocol), and an RTP control protocol. These are also used in streaming.

Streaming can be unicast or multicast. Multicast technology must be supported by network routers, but permits a single copy to be delivered to all clients. Multicast saves network bandwidth, but removes the clients' ability to negotiate flow.

Microsoft's Active Streaming Format (ASF) files are supported by the latest Windows Media Player. Real Networks' provides a free client RealPlayer for its Real Media format (RM). RealPlayer clients are available for Windows, Mac and Unix platforms.

Streamed media players can download a file directly or by accessing a metafile. Metafiles are written using mark-up similar to scripted HTML. They do more than just act as a

translation service; they also provide play and sequencing options. If a series of files were created to suit specific network conditions a metafile could be used to deliver the correct one to the correct client. They also facilitate the creation of presentations comprising several elements some of which can run concurrently.

Pre-recorded Movies.

Movies for asynchronous delivery can be encoded from a live source or converted from an alternative multimedia format. Real encoder supports *avi, mov,* and *wav.* The Microsoft equivalent NetShow Encoder supports *avi, wav* and *mp3* (MPEG audio format). Encoding matches the audio and video compression ratios to specific network conditions and presentation formats. Standard codecs are used. One likely benefit of this is the potential for greater multi-platform support. Sun's Java Media Framework API offers the opportunity for incorporating rtsp- compliant media players within applets or other Java applications. JMF supports Real Systems media.

Metafile-based Streamed Presentations.

It has already been stated that metafiles can be used to assist in the delivery of streamed media. They can also be used to produce synchronised streamed audio and graphics-based presentations. Image files are sequenced, linked to a time line and synchronised to a soundtrack. Other events such as URL links can be included to assist with event automation. Microsoft's NetShow TAG Author and Real Systems' Real Slide Show both generate a streamed slideshow from *wav, bmp* and *jpg* media. Microsoft PowerPoint, is supported by both Real Systems and Microsoft manufacturers. Presentations containing narratives can be converted, synchronised and streamed. Additional benefits include compression. The conversion tools can be accessed from the standard menu lists. Embedded objects, for example animations, do not embed well.

Live Delivery.

Streaming technology has opened up the opportunity for large-scale dissemination of real-time multimedia. Radio and other mass media already exploit it. However it can be also used on a much smaller scale using more modest resources. NetShow includes not only an encoder but also a media server. Using capture sources, such as a microphone and standard web cam, the encoder can send an ASF file to the NetShow Server. Up to 15 clients can access the file directly by making a Media Stream Broadcast Distribution

(MSBD) request on the NetShow server. MSBD is a Microsoft specific protocol. Alternatively a dedicated server can access the stream and serve out to a larger number of clients. Real Systems provides a similar service using its Real Encoder and Real Server products. Once again, platform issues may pay a significant role in selecting the best solution for delivery.

2.4.6 Collaborative Tools

Collaborative networking tools can add real added value to the multimedia network. Email and chat tools are well established, and standards are now emerging for more sophisticated cross platform collaboration. The International Telecommunications Union (ITU) has developed 3 groups of protocols relating to video, audio and data conferencing - the H.320, G.700 and T.120 families respectively. The IETF offers the RTP [IET99], referred to under streamed video above, and the Resource Reservation Protocol (RSVP).

Email

It could be argued that email is not a true collaborative tool, since it provides one way transmission of data. However, it is an effective tool for passing text messages and delivering attached files to individuals or groups.

Chat

Chat is a supports synchronous group communication using a shared text window where participants submit text one line at a time. The text is delivered to each participant in order. Chat applications are not complex, and several commercial conferencing applications include chat as a tool. White Pine Software's CuSeeMe family, Microsoft NetMeeting and Sun System's SunForum all have chat tools. Of these SunForum runs on Unix and NetMeeting on Windows and CuSeeMe on Windows and Mac.

Commercial chat applications use the T.120 data conferencing protocol series. T.120 itself defines protocols and services for Multimedia Data Conferencing. T.134 defines the Multimedia Application Text Conversation protocol (T.Chat). T.120 conformity brings with the opportunity for chat between across applications.

Whiteboards and Application Sharing

Shared whiteboards are group drawing applications. User inputs are communicated to other conference participants in real time. T.120 compliant whiteboard tools are provided

by most commercial conference applications. T.126 defines the MultiPoint Still Image and Annotation Protocol and is basis for screen sharing and remote application control. NetMeeting lets a user share several applications, keeping control or passing it to other participants.

Video Conferencing

Video conferencing applications establish and maintain unicast or multicast real-time audio and visual links between participants. Sharing audio-video resources involves the same quality, network and compression issues as were discussed under audio and video media. ISDN conferences require hardware Multipoint Control Units and are supported by ITU's H.320 protocol; TCP/IP networks require a H.323 compliant server and are supported by H.323 (LAN) and H.324 (Dialup).

The distinction between H.323 and H.324 arises out of available network bandwidth. In terms of video compression H324 limits data rates to 64kbps (H.263); in terms of audio to 6.4kbps (G.723.1). The most visible consequence of these limits is the size of the display area. The Common Interchange Format (CIF) for screen size is 352x288 pixels. H.323 sets a minimum display area at quarter screen (QCIF, 176x144). At the client the actual presentation area will be determined by the monitor settings. H.263 is effectively the video standard for the Internet and dial-up clients.

Leading H323 compliant video conferencing software includes NetMeeting and the CuSeeMe family. CuSeeMe provides for up to 12 concurrent audio-video links; NetMeeting limits it to a single point-to-point connection. CuSeeMe runs on both Windows and Mac operating systems; ClassPoint, a CuSeeMe sibling will also run on Unix and the conference server MeetingPoint already runs on Windows and Unix (Solaris) platforms. Most H.323 clients can connect to the MeetingPoint server, so permitting interoperability

2.4.7 Conferencing Summary

Many conferencing applications are using open standards for real time communications. Chat, audio and video can be provided form desktop applications. The quality of an a video transmission is based upon the available bandwidth. This is most visible in video frame sizes. Conferences can be multicast or unicast. Conference servers are required to host conferences. Servers have cross platform support, clients are more platform

dependent. Application conferencing permits desktops and other applications running on a host to be shared with other conference participants.

2.5 Client Issues

The primary goal of any web-based learning application is to deliver a learning experience to a user or student. Each user is unique and has his or her own profile. Part of that profile concerns the platform being used to access the application. We have already mentioned open standards and how browser manufacturers are undermining the concept. A lack of conformity to open standards impedes the student's ability to access his learning material. Thus, measures must be taken to avoid non-standard content or to adapt it to play on all browsers. This section concerns web clients and how they affect content delivery

2.5.1 Hardware and Operating Systems

The consumer computer market is dominated by Windows and MAC platforms. However, Unix plays a large part in academic establishments. Most applications will not run on all platforms. An examination of the key cross platform specific issues follows.

The Intel - Microsoft Camp

Intel processors and Microsoft Windows operating systems are the main delivery vehicles for the deployment of information technology to the business and consumer markets. Most personal computer and workstation applications can be delivered using them. Most new technologies, in particular with respect to multimedia and collaborative networking, are supported by them. Educational material developed for the Windows OS has the greatest potential for distribution.

Microsoft has provided the means for Object Linking and Embedding (OLE) since Windows 3.1. With OLE, an application can be run from within another application. OLE developed into COM [MSD94], DCOM and ActiveX. Active X controls (binary files) can be registered with a Windows registry. Embedded data files associated with ActiveX components can be executed simply within the Windows environment.

Apple MAC

The Apple MAC is a significant player in the business and consumer sectors. Apple technology is built upon a RISC (reduced instruction set computer) processor and is well

suited to the development and delivery of Multimedia content. Web multimedia technology is well supported by Apple making it a ready vehicle for delivering educational material.

Unix

The Unix operating system can be installed on most hardware platforms. It is used mainly in research establishments or in supporting computationally heavy applications. Its distance from the consumer and desktop sectors has resulted in limited support for Web applications.

2.5.2 Client Software

WWW Client

A web browser requests and formats html documents. In addition it processes proprietary scripting and other non-html data. Application data is identified by a its MIME type, a content definition specification produced by IETF, which a web server includes in the document header. When the web client reads the header it checks the mime type to see if it can handle the file. Certain support comes as standard, other requires subsequent installation and configuration. Multimedia presents a significant problem for web client because of the diversity of file formats used.

Standalone Applications

Standalone application software is built to operate on specific platforms. Some applications will run on several platforms, e.g. Netscape Navigator version 4.61 has Windows, Mac several Unix variants. Microsoft NetMeeting, however, will only run on the Windows family. Applications written in platform independent languages, such as Java, offer the greatest potential for universal use.

Java

Java is the de facto standard programming language for the Web. Tools are increasingly being developed to meet the needs of networking applications. For example, the Java Media Framework (JMF) provides developers with the means to build easily portable applications for capturing, streaming and viewing unicast and multicast audio visual content using RTP.

2.5.3 Fattening the Client - Helper Applications

Helper Applications

Helper applications are standalone applications installed on the client and registered with the web client. On recognising a registered mime type, the web client runs the application and passes it the data file. The application is run outside the browser in its own window. Users should be familiar with installing clients and configuring their browsers.

Netscape browsers can play certain third party applications using plug-in applications. The plug-in software is registered with the browser and runs on top of the browser software. Using a plug-in, the media file can be embedded within and played from an *html* page. Plug-ins may be browser version specific.

ActiveX controls are binary files written to run on Windows platforms. ActiveX controls can be downloaded and registered automatically with the operating system. A web page can reference an ActiveX control[MSD97]. When Internet Explorer sees an ActiveX control is checks its identifier against the registry and invokes the application if appropriate. The control runs outside the Web Client. The output can be displayed within an *html* page. Netscape does not support ActiveX directly, although there is third party plug-software available to provide it.

Java Applets are Java applications that can be embedded within and run from an HTML page. Recent web clients include a Java 1.2 VM within the browser itself. Thus clients can invoke applets seamlessly and pass them data embedded in a web page. As future developments in Java materialise and more players are programmed in Java, the need for proprietary media players will reduce and openness will improve.

2.6 Internet Server Technologies

So far discussion has focussed on the client. Content should be available to meet client requirements; it should also be deliverable. Servers are designed to deliver specific content and to meet specified performance criteria. We have seen that multimedia tends to be data heavy. This section will consider several types of content server.

2.6.1 HTTP (Web) Servers

Web Servers serve *html* and other content to http clients. The Apache Web Server (AWS) is the most widely used web server in the world and is very robust. Although rooted in the

Unix domain, recent binary versions can run on an Win32 platform. The Microsoft Internet Information Server IIS is released as part of Windows NT and is fully integrated into the Microsoft Back-Office package. It occupies second position in terms of web server popularity. Sun System's Java Web Server (JWS), entirely written in Java, is platform independent. Currently, its performance does not match that of the either IIS or AWS. However, it does support servlets technology directly .

2.6.2 Streamed Media Servers

Specialist servers are available to provide most benefit form streaming technology. Streamed media can be served from Real System's Real Server (free version available) and from Microsoft NetShow. The latter, however is restricted to the Win32 platform. RealServer, a commercial server, is a scaleable application and provides greater cross-platform freedom than does its Microsoft equivalent. Live content can be broadcast to large audience and archived on the server for replay, but with varying degrees of latency experience by clients. Initial buffering, lasting several seconds, is also experienced when streaming archived content.

2.7 Middleware Applications

Middleware applications can be used to manage the flow of data between heterogeneous distributed applications. Web clients provide access to Web servers. Web servers can be used to access other resources on behalf of the client.. This section will examine some of the technologies available to assist.

2.7.1 CGI and Servlets

The CGI (Common Gateway Interface) enables user data to be passed from a web client to a server using http Get or Post requests. The server then invokes an handling application and passes it the data. CGI scripts are normally written as Unix shell scripts, although Perl and DOS batch commands can also be used. CGI presents 2 main concerns: scripts and handler applications can dangerously expose the host operating system to hostile user inputs. Secondly, for every CGI request a new copy of the application is called, with an associated overhead.

The AWS uses CGI. IIS can handle CGI, however its preferred mechanism is Microsoft's Active Server Pages (ASP). Further discussion of ASP is beyond the scope of this dissertation. The Sun System solution to the limitations of CGI is the servlet.

Servlets may be seen as server-side applets written in Java. They are invoked by and on a web server. Security and firewall issues are handled by the web server itself. Once loaded, a servlet remains resident in memory and available for reuse. Thus performance is considerably improved. Servlets are supported by the JWS directly and indirectly by several other commercial web servers, for example Apache, IBM Internet Connection Server, Lotus Domino Web Server and W3C Jigsaw.

2.7.2 Database Management Systems

The internet file structure is largely unstructured. Hyperlinks provide a means to navigate between pages, however without defined inter-page relationships, managing the mass of information is difficult. The corporate world has been structuring and manipulating data for a long time. It uses databases.

Relational databases store data in 2-dimensional tables. Because an entity can be referred to in several tables' relationships may be defined. A Relational Data Base Management System (RDBMS) manages the tables and the inter-table relationships. The RDBMS provides a simple mechanism for structuring complex relationships between data items and for fast efficient access to them. Binary data can also be stored, however no data typing is generally provided.

Object Oriented (OO) databases were developed to meet the needs of the OO community. Data is not stored in tables and can be of any type. Data items are serialised and stored. The OODBMS maintains a reference to that item. However, OODBMS were neither as fast nor as efficient as their predecessor, so a compromise was sought.

Object Relational Data Base Management Systems (ORDBMS) permit both the storage of complex data types and of complex relationships. User-defined data types can be established and used in normal data tables. Oracle, the leading commercial DBMS is an ORDBMS. In addition to standard and user-defined types it includes specific support for XML.

2.7.3 Database Access

We have seen that relational databases set out data in tables and that a DBMS manages the relationships between them. Ideally the data could be accessed directly.

SQL

The Structured Query Language (SQL) is a generally standard language for accessing database items. Its format is as follows: SELECT data FROM table(s) WHERE criteria and ORDER according to.. SQL variants are found for most relational and object relational databases with support for user-defined types as appropriate. Applications can be written to invoke database drivers and call SQL statements with a result set returned. Java includes a SQL package in its 1.1 release.

ODBC

Microsoft which had developed its own products Access and SQL Server sought to provided access across databases. ODBC (Open DataBase Connectivity) resulted. An ODBC manager can be used to create a driver giving password access to a SQL compliant database. The Microsoft ODBC manager provides for connection pooling. Both Access and Oracle support ODBC with drivers normally written in C. The degree of access afforded is limited to the ODBC function set, which may be considered significant.

JDBC

Java DataBase Connectivity (JDBC) provides for platform-independent remote access to SQL compliant databases. It uses 4 types of driver:

- Type 1: The JDBC-ODBC Bridge. Java applications connect to an ODBC driver on the same host. The driver connects to the database, passes the SQL statement, receives a result set, then passes it back to the application. There are several stages with translation at each interface. The overhead is significant and possibly unacceptable. However the driver comes as standard, is easy to use and may be the only solution if no other JDBC driver is available.
- Type 2: Java to Native API. Type 2 drivers are supported by native libraries which must be installed along side the driver. The driver can communicate directly with the database reducing the overhead considerably
- Type 3 Java to Proprietary Network Protocol. Type 3 drivers are very flexible and well suited to Internet deployment. They normally communicate via a middle tier, which executes the database access, and tend to be product independent. They can be deployed within applets and have a small footprint.

- Type 4: Java to Native Database Protocol. Type 4 drivers communicate directly and therefore offer the greatest performance potential. However they are database specific and not an open solution.

JDBC gives developers simple mechanisms to make structured data available to users via web browsers. Furthermore it provides for data submission and greater user interaction. Electronic commerce already exploits this facility. However, it can also be used in structuring web sites and in web page delivery.

2.8 Summary

In this chapter it has been shown how technology can be used to create and deliver media content, and at how it can help in structuring information so that information is served selectively in accordance with user requirements.

Current technologies have facilitated the wide use of multimedia, although the diverse nature of the Internet presents obstacles to its deployment. HTML, the basic delivery container for presenting text has developed into one which supports document structure and behaviour. Dynamic pages can be interactive and include multimedia and even data files for client side processing. But this increased functionality has the cost HTML its cross platform standard. XML builds upon document structure, provides a framework for implementing behaviour and is set to succeed HTML. Structure can also be provided by conventional databases. Database manufacturers are providing interfaces so that web pages can created and on the fly. Java, ODBC and SQL are widely recognised tools facilitate this. Thus databases can be used to provide structure to content accessed over the web.

Raw multimedia content is generally large and requires compression prior to delivery. The selection of a particular codec should be based upon network, content and host hardware considerations. Streaming permits the immediate presentation of live or recorded movies at the client and provides mechanisms for creating and managing composite presentations using metafiles. Thus streamed compressed video is the most appropriate audio and video format for web delivery.

Conferencing technology provides the degree of synchronous multimedia and data communication and application sharing necessary for group collaboration tasks. Video quality is limited by the network and is most evident in screen size, frame rate and flicker.

Dedicated conference servers are required for large groups. There is limited support Unix.

Users access media content using a client application. Not all content can be delivered on all host platforms. Most development has been in support of the Mac and Microsoft OS with little support for Unix. Most computer users are familiar with using a web client. Browsers not only present *html* documents, they can also invoke certain application software. Java applets can be executed from within a web client without the need to modify the browser configuration; plug-ins and ActiveX controls must be installed.

Dedicated servers should be used for serving multimedia content. Web server scripting permits client interaction with the server. Java based servers support servlets directly; CGI and *asp* are widely used by other servers. Servlets offer the benefit of cross platform support, security and persistency.

Servlets and other scripts provide access to middleware applications, such as database management systems. DBMSs may be relational, Object Oriented or object relational. The latter is most appropriate for storing complex data structures. Leading DBMS can serve information as *html* or *xml* directly. Alternatively SQL access can be afforded using ODBC drivers and JDBC. Thus access to structured information can easily be achieved across a network.

It can be seen that the there are many technologies which can be exploited for creating and delivering learning material. Audio, video and graphics content can be encoded using cross-platform formats, compressed and served from dedicated servers. Conferencing offers much functionality, but not on a cross-platform basis. HTML can act as a universal container, carrying content and the means to play it to a web client for presentation. There exists, currently, some complexity in fully utilising the potential of *html* in a heterogeneous environment, however scripting fixes exist and Java promises the development of cross-platform players in the near future. Effective open access to this myriad of content can be eased by structuring information in a relational database and providing a Java web interface to it.

Chapter 3 Education

3.1 Introduction

So far we have discussed issues concerning the production and delivery of different multimedia over a distributed network such as the Internet. We have seen that data can be structured and that clients can request specific media as required. But this dissertation is particularly interested in delivering educational content and in doing so to meet specific educational goals. The aim is to provide the student with an educational experience to a predetermined standard. We know what we can deliver and how; we will now consider what is required.

In this chapter we will look at learning theory and its effect on how courses are designed and structured. Then we will consider learning form an individual perspective and question how individual approaches to learning differ. Finally we will consider education as a social activity and examine how collaborative activities play an important role in the educational process.

3.2 Learning

Much research has been carried out into educational theory. Learning theory concerns how people in general learn, how they learn individually and how teachers can assist in the learning process. Course components should have specific objectives which may conform to Bloom's Taxonomy and should be presented in a manner acceptable to the learning ability of the student. [CAR99]. Individual Learning Theory

Learning Theory suggests that individuals learn as a result of encountering some external stimulus. Learning is evident as a change in that individual's performance or potential performance. But there is disagreement into how individuals learn and what specifically are the causes. There are three main schools [DIL98]: behaviourism, cognitivism and constructivism.

- Behaviourism suggests that humans can be conditioned to events and that this conditioning leads to changes in behaviour. In problem solving, humans receive a result and make predictable connections between the problem solving process itself and the result. Repetition reinforces learning and so reinforcement is essential for learning.

- Cognitivism sees the learner as an information-processing organism where the mind takes information in and stores it for subsequent retrieval. Individual perception plays an important role in how the individual's attention is focussed. Thus how information is received differs; similarly how it is translated into something meaningful and stored differs and these vary between individuals. Thus using cognitivism, teachers assist students organise the material and translate it into something familiar and meaningful. Tools to assist include questioning, highlighting, analogy and mnemonics.
- Constructivism extends the concept of individual perception to suggest that not only do individual perceive events differently and incorporate information into their own cogitative structures, but also that the structures themselves develop and adapt. Established structures are applied to new and even apparently unrelated information. Teacher-student collaboration assists in providing the opportunity for students to actively seek out the information and to reflect upon it prior to incorporating it in memory. Constructivist techniques include situated modelling, caching and fading.

3.2.1 Learning Styles

In recognising that individuals learn differently it follows that learning styles can be categorised. From these categories a personal profile can be constructed which defines a student's preferred learning style. New information, which is presented according to the profile, will be well received; that which isn't may face opposition. Felder [FEL] suggested the following categories: Active-Reflective, Sensing-Intuitive, Visual-Verbal and Sequential-Global and proposed strategies for satisfying their needs. Individual profiles are weighted for each category, for example 60% active: 40% reflective. The use of a simple questionnaire can be used to produce the profile.

- Active-Reflective. Active learners tend to retain and understand information best by doing something active with it, discussing or applying it or explaining it to others. They work well in groups. Reflective learners prefer to think about it quietly first. Lectures delivered once in real-time may not afford the opportunity for reflection Note taking with subsequent summary and review may however provide it.

- Sensing-Intuitive. Sensors tend to be practical and patient problem solvers who prefer established methods, facts and details and dislike surprises and abstract concepts without clear links to the real world. Discussion and brainstorming can help to show how known concepts apply. Intuitors, being the opposite, grasp new concepts easily, work faster, though sometimes carelessly, and more innovatively, but dislike repetition, memorisation and routine calculation. Teachers can assist intuitors by binding facts to theories or by asking them to come up with the theories themselves.
- Visual-Verbal. Visual learners remember best what they see. They respond well to diagrams, sketches flowcharts, animation and video. The use of colour coding and concept maps can link key concepts. Most people favour visual learning, but most material is presented verbally. Verbal learners prefer explanations, both written and spoken, and respond well to group-work and lesson summaries.
- Sequential-Global. Sequential learners prefer their material ordered in progressively logical and linear learning steps. Global learners tend to learn in large jumps and often fail to grasp the subject until the big picture is clear. They may be quick and use novel methods in problem solving and connect apparently unrelated topics. They benefit greatly from the use of overviews.

Web based course creators should be very aware of the requirement to produce material focussed at particular categories. They should also be aware of their own profiles and try to avoid subjective design. Material can then be structured according to the needs of the student. It should be noted that not all material should be of the preferred type. In an educational environment students should learn to deal with novel and unfavourable information.

3.2.2 Bloom's Taxonomy

Bloom's Taxonomy [DIL98] sets out to categorise the educational objectives applicable to the classroom and educational institution using three domains: cognitive, affective and psychometric. Within each domain the categories were arranged in a hierarchy with high order representing a greater level of achievement.

The cognitive domain concerns the acquisition and application of knowledge. Its hierarchy is

- C.I. Knowledge: the ability of the students to give evidence that they can recall or reorganise facts, terms or theories presented to them.
- C.II. Comprehension: the understanding of the meaning and the intended use of the knowledge learned.
- C.III. Application: the ability to apply the previously learned knowledge and comprehension in a new situation.
- C.IV. Analysis: the ability to breakdown material into its constituent elements and to see the respective elements' relationship with each other. Also, the ability to reach a conclusion based on the ideas or facts presented.
- C.V. Synthesis: the ability to combine and assemble existing knowledge elements into new and meaningful concepts.
- C.VI. Evaluation: the ability to judge the value of the material using criteria which are both explicit and coherent.

The affective domain concerns the student's attitude and values derived from the institution rather than then the course material directly. Its hierarchy is

- A.I. Receiving: a student's willingness to join the course
- A.II. Responding: a willingness to participate actively in the learning activities.
- A.III. Valuing: a judging of the course in terms of social and moral judgements.
- A.IV. Organisation: the ability to separate values and elements together for the purpose of comparison and relation.
- A.V. Characterisation: the ability to take the organisation level a step further and to construct a coherent value system.

These qualities reflect a sense of being part of the course and are most easily neglected when social intercourse is not built into its activities. Clearly, if the students is remote, achieving the desired degree of intercourse requires imagination. Interaction between students and the teacher can support affective progression.

The psychometric domain concerns motor skills and was not fully developed by Bloom. It is inappropriate to this study and will not be discussed further..

3.3 The Learning Environment

Schools and colleges do more that just deliver a stream of educational material, they also serve as a means of initiation into the social world of the workplace. Teams co-operate on

problem solving, often practical, activities. Learning activities should be designed to develop the appropriate team and practical skills. This requires a conscious and methodical approach to course design. The instructional systems development (ISD) approach advocates three stages of analysis [CHO``99]: Assessment of Needs (why?), Learner Analysis (to whom?) and Task Analysis (what?) prior to the developing a course structure or its components.

Where course delivery will be over a network good method will identify potential areas of weakness. To offer lower learning objectives for online courses than class-based courses is unethical [LAW97]. It is not that online courses require extra consideration in their design, rather that they are more susceptible to poor design if weak method is employed.

Thus, online learning activities though not identical to class-based ones, they should be equivalent. An examination of distance learning may provide a shortlist of important issues. Lawhead proposed that a distance learning environment be defined as follows (quote) [LAW97]:

- The student is separated from the teacher for at least part of the learning time.
- The learning is influenced by an educational organisation that has planned and prepared the learning material and that supports the learning process.
- Technical media print, audio, video or computer) are used to unite the student with the teacher and with other students, as well as with the content of the course.
- The student can use two-way communication to benefit from or even initiate dialogue.
- The student has the responsibility for his learning process.

Most students would consider that this definition could apply equally to a modern campus-based educational environment. Thus although the student is responsible for progress, the institution bears the responsibility for providing a sound learning environment including support throughout the process. Great weight is given to the use of two-way communications in uniting the student body. The report suggests that the Web offers the means to overcome many of the difficulties associated with traditional distance education, in that much of the content flow was one-way and received passively at a predetermined rate. But that it is not the technology itself which is key, but the adoption of a strategy to engage the students in interactive learning activities.

3.3.1 Individual Learning

We saw earlier that individuals have their own learning profile and that ideally learning material is compatible with it. Lesson content should be designed to meet specific learning needs and available to the appropriate profile group. Given the range of available profiles there is a danger that too much expensive material is assembled and that the student is overwhelmed. Only that which is most appropriate to the student at a particular time should be selected and provided for him.

Hypermedia research has focussed upon how structure can be applied to information and how that structure can be used to help people work with the information [NUR]. Adaptivity is used to modify the structure or appearance of the information delivered to suit the profile of the user. This may be to restrict the information made available to the user or to provide mechanisms to assist in navigating through it.. Most systems make a profile record of the student, a profile for the course and make material available as necessary. Others use databases to generate course material on the fly.

An adaptive hypermedia course (CS383) developed at the USMA used tailored the presentation of learning material in accordance with the students' profiles [CAR97]. The profile was based upon Felder's preferred learning styles. It was found that adaptivity aided navigation and improved learning effectiveness. A summary of the criteria is given in figure 2.

	Lesson Objectives	Slide Shows	Hypertext	Response System	Digital Library	Media Clips
Global	X		X	X	X	X
Sequential		X	X			X
Verbal	X	X	X		X	X
Visual		X			X	X
Active		X	X	X		
Reflective	X	X	X	X	X	X
Sensing		X	X	X	X	X
Intuitive	X	X		X		X

Figure 3.1 Summary of learning styles and hypermedia course components.

Material which does not conform to a student's profile will not be learned as effectively as that which does. This applies equally to the class room. The provision of alternative sources of material along the lines of that shown in figure 2 would also be of value to the student who does not respond well to the class room. Thus once again, methods and structures necessary to support online or distance delivery are equally relevant to the class room.

3.3.2 Group Learning

Collaborative Learning

Formal collaborative learning includes group work, tutorial sessions, discussions and presentations. Informally collaboration also plays a part in lectures in the form of questions and answers.

In the class room the teacher has immediate and direct student feedback and can gauge how well his material is being received. He has the power to adapt the form of delivery through using emphasis or other methods to ensure that the key points are absorbed by the class. Students use questioning not only to seek new information or to resolve misunderstanding, but also reinforce what they have learned or to adapt what they have just learned to fit their own learning structures. Questions from other students and small group discussions provide help in this constructivist process. For online learning environments it is also desirable to provide interactivity and student feedback. But for the feedback to be an effective control mechanism, it too should be as immediate and direct as possible.

Group-work satisfies Bloom’s application and analysis objectives and supports multiple learning styles. For it to be effective it should include the following elements [HOW96]:

- Group Processing - task allocation, progress monitoring and co-ordination of interdependent tasks is managed by the group .
- Individual and Collective Accountability - each member should be allocated specific tasks but is also responsible for the overall performance of the group.
- Appropriate use of Collaborative Skills - collaborative skills are not purely technical; they include trust building, decision making, conflict management, leadership and communication skills. These soft skills are most easily neglected in

the scientific and engineering educational sectors, but are highly valued by employers.

- Face-to-Face Promotitive Interaction - individual tasks are judged in terms of the success of the group solution, not individually.
- Positive Interdependence - group success is judged by its weakest member.

These elements require close co-operation and are most easily satisfied by small groups in a face-to-face environment supporting a shared work area. But established social stereotypes and individual personalities can impede the effectiveness of the activity; a mature student with work experience can easily dominate group discussions. Online groupwork can be monitored by the tutor. The security afforded by the physical distance between group members can encourage the more reticent students to play a fuller part in the overall group activity [LAW9]. Thus, online collaborative activities can add value to the traditional class-based group assignment.

Social Learning

Graduates are judged not only on the technical skills they have acquired at a place of learning but also on acquired social attributes. Bloom addressed some of these issues within his affective domain. At first the student consents to be member of the course, but as the group forms, consent develops into a sense of being a part of the course and subsequently, perhaps, into wanting to influence its development and that of the institution itself. Sports facilities, coffee breaks and other recreational activities support this progression. The institutional culture which results may become as much as stamp of the course's worth as the academic achievement itself.

3.3.3 Distance Learning

Distance Learning was defined above. Lawhead developed the definition using proximity in terms of time and space. The classroom lecture, with immediate delivery before the class can be defined as near in time and near in space. On the other hand, the traditional distance learning package may be considered as far in both time and space. But what of the use of course notes on a college web site, the submission of assignments using email, the use of campus networks for routine inter-student message delivery, chat, discussion groups and notice-board information. Clearly, the use of distributed technology in support of education is not exclusively restricted to distance learning;; it is already a reality on

campus. Thus distance education is not restricted to organisations like the OU; it applies to all institutions.

3.4 Summary

In this chapter we saw how learning theory is supported by three main schools: behaviourist, cognitive and constructivist. From theory, it was deduced that the role of the teacher is to facilitate learning though directing activities at individuals or groups. Individuals respond better to certain information formats and types of activities; groups not only provide useful team work skills, they also support Bloom's higher order objectives and provide support for alternative learning styles.

Distance education course design was offered as an example of where structured method, for example ESD, is essential if quality learning is to be achieved. It was argued that as technology plays a greater role in campus-based courses, similar structured methods should also be applied there. Method should support the development of learning components which satisfy specific learning objectives and which cater for the needs of individuals and groups. Adaptivity should be used to limit the quantity of material offered to the student and to assist in its navigation.

Collaborative learning activities satisfy Bloom's objectives and support different learning styles. They also develop important social and work related skills. The physical distance afforded by online delivery can encourage fairness by forcing less dominant personalities to play an active part in group tasks. Thus delegation is more likely and individual and collective responsibility is reinforced.

There is a now blurring of the distinction between distance and campus-based education. As technology forces itself onto the campus and distance educational institutions embrace it as a means of improving efficiency and effectiveness shared common teaching methods will result. The discipline required to develop distance courses will adopted on campus.

Chapter 4 Delivering Education

4.1 Introduction

We saw in the last chapter that the design of educational courses, whether online or class-based should be founded on sound pedagogic theory and that a methodical approach can be applied to curriculum development. Structured method implies management control and openness in the development process. Learning components should be defined to meet specific objectives and should satisfy different learning scenarios. Components should meet the needs of individual students and also those of groups. Collaborative activities should be offered. Online delivery may enhance certain group activities. Component design is a creative process.

In the previous chapter we looked a many different technologies. In a distributed environment platform and network bandwidth issues influence greatly. HTML is a universally accessible page format deliverable over most computer platforms, but it is primarily focussed on presenting static text based content. Increasingly *html* is being used to deliver dynamic content and data for client side processing. Database technology can be harnessed to produce pages on-the-fly so permitting adaptable access to complex information structures. Compression is required if multimedia is to be delivered effectively and this can be further improved using streaming. Compression and streaming can use open-standard or proprietary formats. Of the latter, certain formats are sufficiently universally supported as to be quasi-open.

In this chapter we will examine how some of these technologies could be applied to delivering education. The process will involve course development, user profiling, content creation, content storage and retrieval and the delivery of learning components. Delivery concerns selected static content as well as support for stand-alone and handler applications.

4.2 Course Development

Course development methodologies such as Instructional Systems Design (ISD) [CLA99] aim to produce courses which consistently meet the requirements of a curriculum. They structure the development process into a series of activities such as analysis, design,

develop, implement, evaluate and lends themselves to being supported by collaborative and project management tools, such as Lotus Notes

The design stage will identify the requirement for a series of learning tasks which may be grouped in lessons. A lesson will have an objective, a means of evaluation and a series of activities or learning components which the student should complete to reach the achieve the objective. Learning components should be developed to satisfy both learning objectives and provide support for individual learning styles.

The use of IT to support course development is not a requirement. Experienced teachers will naturally incorporates a variety of learning activities into their lesson plans. Teachers may resent an invasion of technology into their academic domains. However, formal method can be used to set quality standards may require that learning components be specified.

The Uppsala grid [LAW97] categorised a series of learning components which could be used to create online lessons. It provides a mechanism for allocating certain media to the components which can be used to implement the components. The table is shown in figure 3. It is neither exhaustive nor the base minimum, for example, no pre-course text is specified.

By extending the grid into a third dimension, objectives, the course designer can specify precisely what goal each component is required to achieve. Course design itself requires to be structured. Consider an example lecture component, below. A scenario will be given, followed by a listing of multimedia components. The objectives listed are those of the course designer.

> The purpose of the lecture is to draw attention to the key lesson concepts. A lack of student understanding should be identified, hence feedback through interactivity is required. The main lesson concepts are applied in every day life as can be illustrated using a short using film-clip. They can be developed and set in specific contexts using animation and diagrams. A possible mechanism for analysis will be presented using a case study. From this the following technology requirement can be identified:
>
> - Hypertext will be used to set the scene, to impart the basic knowledge and to develop it in a logical and sequential fashion. Hyper links will be used to connect with supporting and related material. Administrative details relating to the lecture will be provided.

- Compressed video. The lecture will be given before a live audience, providing realism and feedback. Remote participants must be able to submit feedback, reinforcing a sense of participation. Sound and video quality should be good. All students should be able to view the lecture synchronously, and review it later.
- The mechanics of a protocol will be presented under various conditions of failure. Animation will be illustrate the dynamics.
- A video clip will illustrate the how people use protocols in every day life.
- The lecture should be multicast synchronously. Text chatrooms, audio or video conferencing is required for student feedback. The lecture should be recorded for review.
- A whiteboards will present diagrams in real time. Data conferencing is required with support for application sharing.

Tools and technologies	Lab	Lecture	Discussion	Simulation	Group work	Assignment	Presentation	Tests
Plain text								
Hypertext								
Hypermedia								
Interactive text								
Text chatrooms								
Bulletin boards								
GroupWare								
Sound								
Voice								
Internet phone								
Audio-graphic								
Animation								
Visualisation								
Quick-time video								
Real-time video conference								
Compressed video								

Figure4.1 The Uppsala Grid, a lesson component development framework. Source: Lawhead et al.

Good web-based course material is very expensive to produce and requires skilled resources. Approval procedures may form a part of the course development process. The

Uppsala grid and similar frameworks can assist in ensuring that the mix and quality of its content is appropriate and achievable. Having defined the course components, those definitions could be included in a database. Search techniques may identify components already implemented using media which satisfy the requirements, so avoiding the need to implement afresh, saving time and resources.

4.3 Student Profiling

Teachers mentally profile their students and adapt the lesson delivery to reach all members of the class. In an online environment, students may be profiled according to previous knowledge and experience, preferred learning style, hardware and browser types, campus or remote or other important qualities. In "Courseware on the Fly" [DIL98], student profiles were based upon previous knowledge and were updated as the course progressed. Profiles were stored in a relational database. AHA! [DEC98] generates a profile file for each student based upon previous knowledge and link navigation preferences. In both applications, the profile is used for selecting the lesson content to be offered to the students. They will described in greater depth in the next chapter.

Profiling should could also include platform and network information. For example, if the student is using a Unix host, content containing unplayable media such as Quick Time movies will not be offered. Host OS and Web client information can be gathered automatically and stored in the user profile. The entry page for Falkirk College, Scotland [FAL99] uses Java scripting to obtain this information and then automatically redirects the client down the appropriate path. Thus, some of the content handling problems discussed in chapter 2 are hidden from the user. From a distributed technology standpoint this enhances transparency.

4.4 Creating Online Content

This section will consider how multimedia content can be incorporated into an online course. In chapter 2 *html* was identified as the best carrier vehicle of multimedia content, on account of the flexibility it affords due to its openness and generally cross-platform capability and its suitability for operating in a distributed environment.

4.4.1 HTML for Text and as a Container

It has been shown that HTML supports dynamic content, including HTML animation, and can act as a container for delivering applets and other data files for client-side processing. Applets run cross platform Java applications. Developments in Java are continually providing greater functionality and are expected to permit the development of client side applications capable of supporting all of the multimedia technologies described in chapter 2. JMF is particularly interesting in this regard. That state of development has not yet been reached. However within a Windows environment, support is provided by plug-in applications and ActiveX Controls in particular.

Today's primary presentation tool, Microsoft PowerPoint, can convert presentations automatically to *html* according to user requirements, for example within a frameset including an index, slides, navigation buttons and notes. The package is created and put into a single directory for delivery over the web, using the default index.htm for simple access. Plug-in tools can also convert presentations to streamed audio-visual *asf* or *mv* format. Thus, teachers familiar with PowerPoint can produce simple but effective web content. PowerPoint converts its animated vector graphic slides to .gif format. Unfortunately, slide animation is lost and launching OLE or other embedded objects is complicated. Thus, alternative vehicles should be also considered.

Recent HTML editors simplify web page creation. Macromedia Dreamweaver, an HTML/DHTML/XML point and click editor with support for integrating not only text and images, but also streamed multimedia is such an example. Its HTML tools construct *html* pages containing embedded third party objects such as *rm* content. Not only is the *html* page itself constructed, but so are the associated meta files. Dreamweaver also supports ActiveX controls, so certain other applications, for example Microsoft NetMeeting can also be embedded with ease. The capability of HTML and its tools support makes it the most appropriate authoring medium.

However, not all applications use ActiveX controls or browser plug-ins, for example White Pine's CuSeeMe. As a result, it is necessary, install the applications on the host as standalone applications. Web servers and clients can be configured to launch the application automatically however using the application MIME-type. In due course Java certainly extend sufficiently to overcome these problems.

Having set the content within an *html* context, it can be accessed simply using its URL. Alternatively is can be generated on the fly using a middleware solution.

4.4.2 Embedded Content

In discussing embedded content it is assumed that *html* will be used as the carrier. Embedded content normally comprises image, applet, video or animation files. The development approach should start with consideration of the student's client application. Then one can identify the appropriate file format, the requirement for format conversion and finally the authoring tool to create the native media file. In this section, we will concern ourselves with video and animation.

Animation

Consider this scenario. It is intended to deliver a simulation in the form of an interactive animation to a student using Unix client. It follows that the final form should be formatted as a *jpg*, animated *gif*, *rm* (version 5) or *swf* (version 3) or *dhtml*. The lossy nature of *jpg* does not lend itself to animation, animated *gif* files do not rescale well, neither support interactivity and *rm* files an inappropriate delivery format for animation. DHTML animation based on layers can be created using an HTML editor such as Dreamweaver, although the animation cannot be subsequently embedded into another page.

SWF animations are authored using the Macromedia Flash application. The animations are interactive and permit the user to stop, pause and continue the animation. Text boxes can be added to provide descriptions or what the animation is simulating, or audio narrative can be streamed with the same purpose. Flash animations use vector graphic. As a result, their file sizes are small and they resize on the screen without distortion. Flash uses drag and drop technology and is quick and simple to use in skilled hands. The application exports its animation as a *swf*, which can be embedded in a *html* document. The process seems very simple.

Streamed Video Content

A similar methodology can be applied to developing a video clip to that for animation. However, in the case of video, processing comprises capture, editing and format conversion. Tape is the most efficient storage medium for raw video data. If several versions of a digital video clip are required, for example to cater for different network

conditions, then working from tape may be necessary and technical assistance should be sought .

Live video is captured and encoded immediately in the correct format and is consequently simpler to produce. However only a single streaming format is possible, so the lowest common denominator should apply. The quality of the final image will be influenced by the camera, its encoding parameters and network bandwidth conditions.

Once a camera is installed onto the host, the process of capture, encoding and delivery to the media server is straight forward. The encoder should be on the same host as the video capture device.

Multimedia Skills Base.

We have seen that there are many tools available to help in the creation of multimedia content and that once created, it is relatively straightforward to deliver it embedded in a *html* page. However, it should also be apparent, that a good understanding of platform and delivery issues is necessary if the process is to be carried out efficiently and effectively. In unskilled hands the creation process can be slow, or the results unpredictable. Thus institutions planning to make greater use of multimedia technologies should consider maintaining the appropriate skills base and should seek to reuse content.

4.5 Content Storage and Retrieval

In traditional file storage, files are grouped in directories, often in hierarchical tree structures. The file structure itself has meaning and can be used to assist in locating a file. Location is significant to the user. Once a user is in an hypertext system, the physical file location, or that of a document component is largely transparent to the user. Hyperlinks connect documents and the system of hyperlinks provide the often complex structure. Architectures are required to support the structure and engines to facilitate search and retrieval.

Well structured hypertext liberates the designer from the need to provide meaning through file structures and permits the scaleable use of flat files.

4.5.1 Flat File with Static Hyperlinks

In a flat file system, the file structure may be independent of the link structure. Assume, a course contains certain information which the course designer wishes to impart to the

students. We will call it the information domain. The designer can distribute the information over several pages and can structure the hyperlinks so that by following all the links, all the information will have been seen. A static link structure can ensure that everyone gets all the information.

As the information domain grows, so does the link structure complexity. Maintaining its integrity can become increasingly difficult. This is especially the case, when links point to locations outside the designer's own domain. The complexity also impacts directly on the user. "Hypermedia applications generate comprehension and orientation problems due to their rich link structure" [DEB99]. Without others means for discovering the semantic meaning of a link, the user must rely on the hypertext author's description. Users benefit from external assistance in navigating complex hypertext. Hence, there is the requirement for hypermedia engines.

4.5.2 Adaptive Hypermedia

Adaptive Hypermedia attempts to assist users navigate an information domain using an hypermedia engine. Most use a user model and adapt both document content and its links to fit. Until recently engines have been largely proprietary, although research is being conducted into developing architectures including implementations for Open Hypermedia Systems (OHS) [OHS95] .

Debra's AHA! hypermedia engine, introduced earlier in this chapter, [DEB99] generates page content and links from *html* fragments using Java servlets, scripting and user profiles recording knowledge and concept information. The main area of adaptivity concerns link navigation. Content written in accordance with its authoring guidelines are be stored as *html* fragments in the root directory. The AHA! engine adapts the fragments and serve them as required. AHA! is used practically in delivering courses at the Eindhoven University of Technology [DEB98]. Being Java and *html* based, AHA! and its courses are multi-platform and easily portable.

AHA! and similar system offer great potential, however they rely upon careful authoring. Contents should not be changed without first considering the impact it will have on the overall link structure.

An alternative approach is to store metadata and link information in a relational database and give client access to it using ODBC. The Southampton OHS configuration [MIL] employs XML and CORBA, using Java RMI. A resource request is invoked by the

client; the server queries the database, obtains the links to the content and passes is back to the client. Conlan did similarly [CON99] using XML(IMS) metadata as the basis for the query.

4.5.3 On-the Fly Generation

Just as the teacher adapts his/her delivery to respond to student needs, it should be possible to make changes to a hypermedia course, so that the course creation is jut-in - time (JiT). Server scripting and other tools permit JiT courses to be delivered on-the fly.

Full generation

Dillon [DIL98] incorporated course content and structure within an Oracle database. Applets would interact with the database and present page content. Courses were composed from sections which comprise units. Content was stored as pagelets and meta information provided a means of matching pagelets to user profiles. The database stored course information in 5 tables.

- Course makeup
- Section makeup
- Unit makeup
- Pagelet information
- Pagelet meta-information

Pagelets matching a user profile query would be formatted and returned to the client for presentation. Dillon recognised the limitations associated with storing large binary data such as multimedia content within the database. Instead it was stored externally. For multimedia content the database would serve a URL link instead of the content itself.

Link Serving

Large databases take longer to query than small ones. A remote query opens a connection which is not closed until the result set is returned. Faster queries permit more connections. So, keeping the content itself out of the database is attractive. There is also an overhead associated with the requirement to parse the output from a database before it can be converted to *html* . Web servers are designed to deliver html at high rates of performance. We have seen that hyperlinks are the main instruments of structure within hypertext.

Excluding content from the database would result in the requirement for an additional connection to a web server, however, arguably, this would still be much faster.

Serving only structure in the form of hyperlinks from a database makes provides consistency and location transparency. The actual content can be served from the most appropriate server, be it *html*, compressed video or any other accessible using a URL. Were that database also used for course development and student administration it also provides a single point of control for managing the entire Courseware solution.

4.6 Summary

In this chapter it was suggested that collaborative work tools be used to help in the course development process. Designer's Edge was introduced as a proprietary application to assist with the analysis and structural development stage. It was argued that a relational database form the basis for a similar tool. The result of the analysis would be a list of learning components and the Uppsala Grid provided a useful framework for categorising and linking them. Well structured content simplifies the application of adaptivity to their delivery process. Adaptivity permits the controlled delivery of information conforming to the needs of the end user. Such criteria would be defined in a user profile and could include learning as well as platform information.

It was suggested that *html* be used as a means for presenting text as well as for delivering other multimedia data. The ability to embed objects simplifies the authoring process, although the lack of a common standard adds some complexity and limits the potential for distribution. The absence of a standard has resulted not only in an inability of clients to handle embedded objects consistently, it has also resulted in a limited availability of cross-platform multimedia players themselves. The complexity which result from this increases the need for specialist multimedia resources.

Three adaptive approaches were offered for storing and accessing *html* content. First, content could be written as *html* fragment files with links and presentation tags written into the hypertext. The hypermedia engine could then load and process the appropriate fragments before serving them to the user. This approach suites static content. Second, the content could be stored in a relational database together with metadata. Users would query the database using metadata. The engine would receive content which it would translate into the appropriate format for the client, for example *html*. The final approach is similar to the previous one, except that instead of content, the engine would serve links.

The final approach would impose a much smaller load on the database. It is more suitable for delivering multimedia content.

A relational database could form the basis of an integrated course development, administration and delivery application by serving adaptive links to selected *html* pages carrying a wide variety of multimedia content.

Chapter 5 The State of the Art in Computer Assisted Training and Education - What It is and What It Should Be.

5.1 Introduction

The research so far has shown that courses should be structured, that lessons require objectives, that presentation is best adapted to the needs of the individual and that group activities should be supported. Current technology is capable of satisfying these requirements.

Several commercial applications are already available, which promise to deliver comprehensive solutions. Others are emerging from the research community and have been introduced previously in this document. The IT support for education covers a wide domain: from the use of computer controlled teaching aids, such as smart white boards and projectors, through skills focussed CBT packages developing personal vocational skills, like typing and telesales, to sophisticated educational environments used to deliver the virtual class room.

This chapter will present an examination of a selection of software applications. It will begin with an evaluation of the functionality provided by several commercial courseware solutions. Then two applications from the research community will be presented which offer an alternative approach to handling course structures. Off-the-shelf multimedia authoring and conferencing applications representing the tools available to the teacher will be described. Two sample courses will be offered to illustrate how multi-sensory content can be combined in practice. A synthesis of the material presented in the preceding chapters will draw the research to a close. Finally a statement of requirement for the development of a unified architecture will be presented. This will be the basis for subsequent implementation.

5.2 Evaluation Criteria

Educational software application can be evaluated as a productivity tool in the hands of designers, developers and managers or in terms of the learning content they produce. A

recent survey [INF98] of leading software applications supported the former approach and selected the following as its criteria:

- Simplicity of content creation
- Flexibility of deployment methods
- Strength of course management capabilities
- Breadth of reporting tools
- Cost of ownership.

The absence of learning criteria is concerning. A different set of criteria could emerge from the second approach and may include [WAD98]:

- Tools for integrating a wide range of interactive multi-sensory experiences
- Support for student experimentation, simulation, auto-correcting testing modules, checkpointing and managed note taking.
- Support for cognitive structures
- Provision of synchronous communication and collaborative workspaces
- Interface quality
- Degree of integration within curriculum development systems
- Facility for integration with faculty support and administrative services

It could be argued that the former are most appropriate for applications delivering training in a corporate environment, whereas the second suits more the educational community. However, perhaps a unified approach would be better as a basis for developing marketable educational applications, targeted at both corporate and educational clients. This will be the approach taken for the evaluation below.

5.3 Commercial Courseware Applications

In considering applications offering solutions that build, deploy and manage on line courses, research showed that “most vendors focus on or two aspects of the Web-based training market" [INF98]. From the corporate point of view, the optimum solution would combine the “course-creation and deployment capabilities of Asymetrix, merge them with

the course-management features of WBT Systems, and top it off with the reporting capabilities of Macromedia."

A closer examination of these packages will provide an insight into how applications are using current technology to satisfy the requirements of the corporate training sector. The findings are summarised in figure 5.1

Evaluation Criteria	Authorware	Pathware	Toolbook	Librarian	TopClass	WebCT
Simplicity of Content Creation	icon flow		page layout		page layout	page layout
Interface Quality	X	X	X	X		X
Tools Palette for Multisensory Content						
Hypertext	X		X		X	X
Multimedia						
Import from external tool	X		X		X	X
Internal authoring tools	X		X			X
Video Capture and Encoding	X		X			
ActiveX	X		X			X
Plugin	X		X			X
Java	X		X			X
Synch Communications and Collaboration						
Email	X	X		X	X	X
Threaded Discussion Groups				X	X	X
Chat				X		
Whiteboard				* * *		X
Video Conferencing				* * *		
Application sharing				*		
Student Activity Tools						
Test pages	X			X	X	X
Experimentation						
Simulation						
Support for Cognitive Structures						
CheckPointing		X		X	X	X
Note taking						X
Search				X		X
Adaptive Structures/Presentation				X	X	X
Review				X		X
Glossary						X
Course Management and Reporting	X	X		X	X	X
Systems Integration Capability						
Internal course development tools	X	X	X	X	X	X
Internal student and course admin		X		X	X	X
External ODBC Database Integration	X	X		X		
Flexibility of Deployment			X			
Proprietary plus plugin			X			
HTML Format	X		X		X	X
CD-ROM			X			

Figure 5.1 Commercial Courseware Evaluation

5.3.1 Macromedia's Attain Enterprise Learning Solution

The Attain Enterprise Learning Solution comprises three components: Authorware, Pathware and Dreamweaver.

5.3.2 Authorware.

Authorware is a course production tool utilising the intuitive iconic flow paradigm (figure 5.2) It supports user interactivity in the form of navigational control and support for multiple choice tests with automatic feedback. Course pages can be created using the tool directly and third party multimedia content can be imported via its library facility. Components are dragged form the library and dropped into the structure. Behaviour and other attributes can then be modified by selecting an icon from the toolbar. Authorware provides templates for many common course components such as containers for embedding multimedia content, test pages and navigation aids. This simplifies the course development process and facilitates the creation of high quality user interfaces.

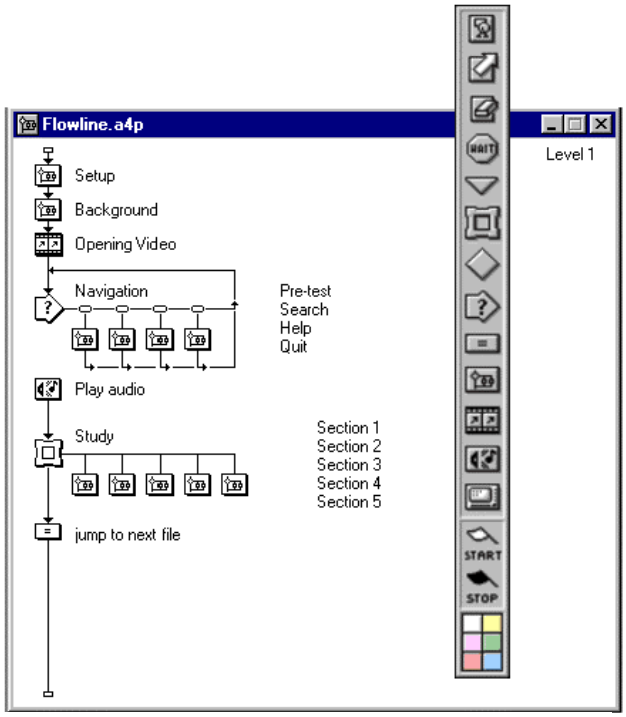

Figure5.2 Authorware Iconic Flow

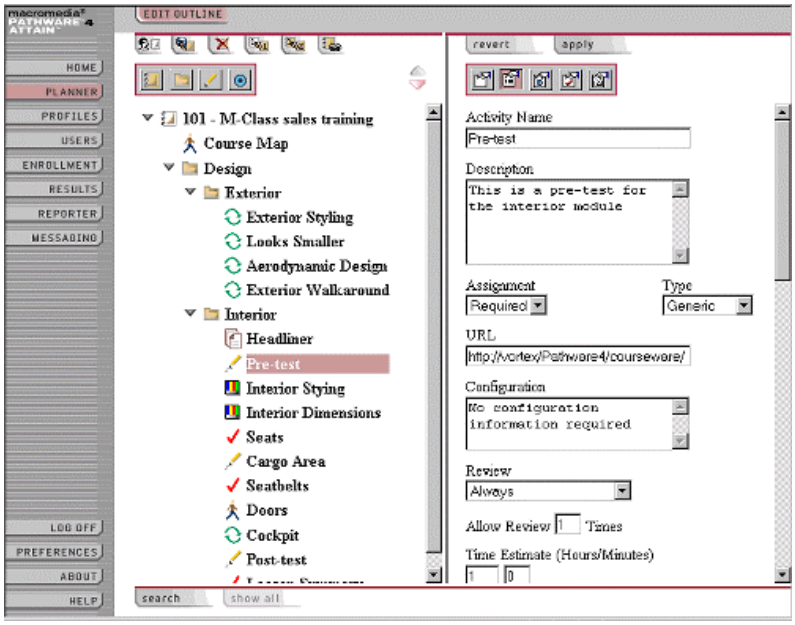

Figure 5.3 Pathware Course Builder

Pathware.

Pathware is a planning, development and administration tool. It uses a web client interface (figure 5.3). Its course management facilities include class and student allocation, course structuring, and student tracking.

Pathware requires that students log on to a course. Check pointing permits the student to leave and rejoin a course module mid flow. It also allows a student's progress through the course to be monitored by training staff. Test results can be collected and student reports generated automatically. Pathware does not support synchronous communication directly although it does provide a SMTP compliant server for managing email communication. The Pathware server supports ODBC and can be integrated with other administrative support systems.

5.3.3 Dreamweaver.

Dreamweaver is an HTML editor with a point and click interface. It has support for support for scripting, *dhtml* components, Java applets, ActiveX controls, streamed media and other page elements. It is a powerful tool capable of producing high quality *html* interfaces with embedded interactive multi-sensory content. In figure 5.4 a the behaviour tool is being used to bind a navigation button to a JavaScript function in the parent document. The tools are easy to use, especially when the content comes from another Macromedia applications.

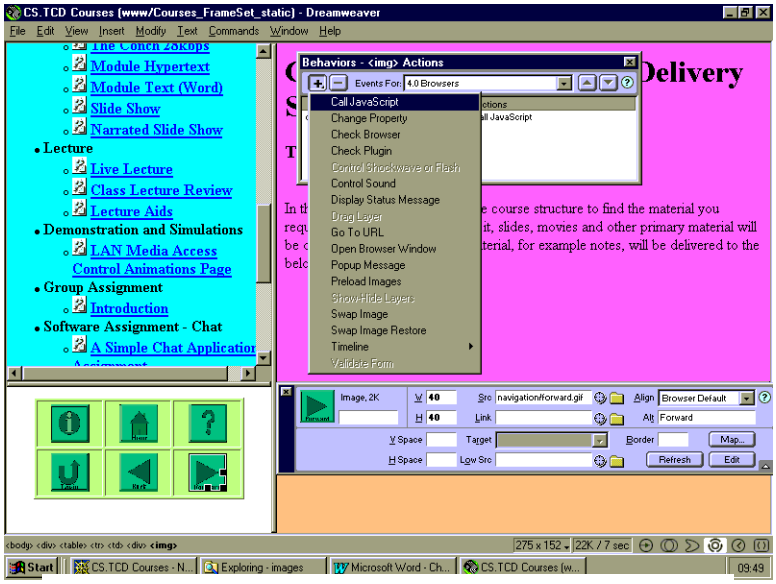

Figure 5.4 Binding a button control to JavaScript using Dreamweaver

5.3.4 Asymetrix Toolbook and Librarian

The Asymetrix Learning Systems' solution comprises 2 components: Toolbook and Librarian. Toolbook provides for course creation, structuring and deployment and comes in 2 versions: Instructor and Assistant. Librarian is the course management and administration tool.

5.3.5 Toolbook

Toolbook, the course development tool, builds courses as a sequence of pages. Content is written to each page using a page layout metaphor. Like Authorware, the library of standard templates is extensive and includes controls for interaction with the Librarian tool. Figure 5.5 shows the catalogue of movie players in front of a course page. The designer drags the appropriate player onto the page then set its properties using another window. Instructor has its own scripting language, which provides the developer with full control over content flow and event handling. Sequencing is provided by page links, These may be hard coded or event triggered. Toolbook Assistant does not support scripting although its GUI interface is simpler to use. Thus it is more appropriate for non-programmers.

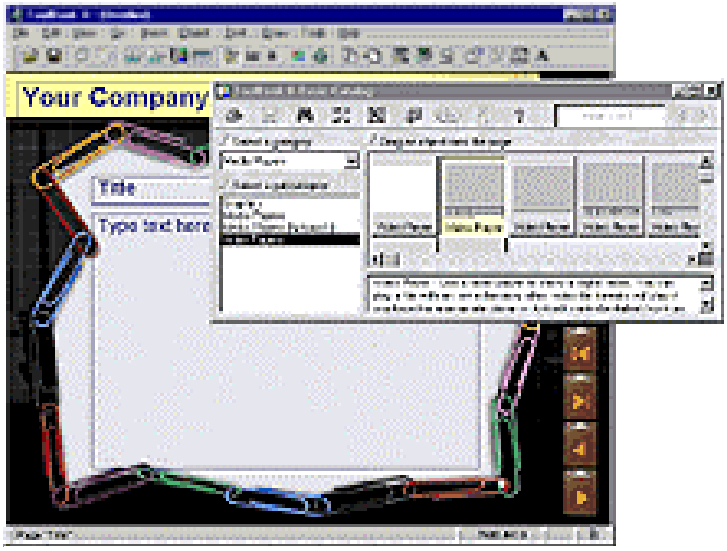

Figure 5.5 Dragging controls to the canvas in Toolbook

Courses developed using Toolbook can be saved in their native format and viewed using a web client plug-in or Windows viewer. Alternatively they can be exported as *html* for open access over the Internet. Network and CD-ROM deployment are supported.

5.3.6 Librarian

Librarian is the course management tool. It uses a database to store course structure and can integrate course blocks developed by Toolbook or other third party applications. It does not support integration with an external database.

Learning content may be located anywhere, for example on a Web server. The Toolbook "Publish to Librarian" tool, creates an interface to the Librarian database automatically. Librarian does provide a curriculum development tool. Wizard based, it guides the course developer through the process of creating a course. It poses questions then

automatically builds the course to specification. Wizards simplify the process of course development, however also the sophistication of the result.

Management utilities include student, group and class administration. Administrative tools provide for assigning groups to activities, progress monitoring, evaluation and reporting. Cognitive structures are supported: course modules may be sequenced in a simple order, or in accordance with specific prerequisite criteria. Synchronous communication and collaboration is limited to email, chat and threaded discussion groups.

5.3.7 WBT Systems' Top Class Server

Both the Macromedia and Asymetrix solutions are tailored to delivering training solutions where skills can be defined and tested. Top Class Server [WBT], on the other hand, emerged out academia and focuses on student teacher interaction.

Top Class Server concentrates on course development, delivery and administration. It is a structural tool. The interface uses a web client. It does not support sophisticated drag and drop content creation tools; updates and navigation through the tool can be frustratingly slow. Courses are developed by adding modules to a hierarchical structure. Modules can be copied or moved within the hierarchy. Learning components are presented as *html* .

Content is best created using third-party applications then integrated into the course. This is not necessarily a drawback since interface development is often done most effectively using a dedicated authoring tool. The tutor can, thus, concentrate on course structure and the interactive aspects of teaching. Conversion tools are available for integrating content created using PowerPoint, Word and other editors.

Learning Content, course structure and administrative information is all stored in the server database. Lesson pages are generated on-the-fly with updates immediately available to students. Interactivity tools include email, discussion lists and class announcements. Course structures can be tailored to groups or to individuals. The tracking and reporting tools are comparatively basic. Integration with an external database does not come as standard.

5.3.8 WebCT

Like TopClass, WebCT emerged out of the academic community. It too focuses on the development, delivery and administration of courses and uses a web browser as the client for all interactivity.

Course content can be created using any HTML editor and uploaded to the WebCT server. A simple text editing tool lets the designer modifying pages from within the application. Multimedia content can be included, however it is most easily accessed using the audio and video tools provided. Embedding controls within a page makes for easier reading but the implementation can be complicated. Course designers and content developers must be prepared to work within the functional constraints defined by the WebCT architecture. This is a disadvantage.

WebCT can be configured to run within both an Apache and IIS server domain. Content outside the server domain can also be accessed using its URL. The WebCT management system uses Perl at its centre and HTML scripting to convert course structures to Web pages. WebCT courses are supported by an extensive set of designer and student tools. The course structure is hierarchical, with an home page at it root and paths descending to its tools or course content. Content is arranged in static page sequences. Course presentation uses frames, so easing navigation across the course hierarchy. Considerable use is made of icons. Course tools include quizzes and self tests, glossaries, indexes, search, content-sensitive test, tips, book-marking and collaborative work areas (chat, white-board, discussion groups). Access to all content mapped to the course structure is tracked. Course administrators can monitor hit rates and other criteria.

The WebCT database does not support integration with other relational databases. Data can be imported, however, from delimited formats such as CSV. The potential for interoperability with other parts of the educational information infrastructure is consequently limited.

5.4 Courseware Solutions from Research

The applications discussed in this section are examples of adaptive hypermedia applications. They were introduced in previous chapters. In this section the focus will move to their functional aspects.

5.4.1 Adaptive Hypermedia Architecture (AHA)

AHA is a generic adaptive hypermedia engine which uses link adaptation. It is built upon the Jigsaw Web Server developed by W3C. AHA is used for delivering several courses at the Eindhoven University of Technology [DEB98]. Being Java based, the AHA engine is easily portable to other platforms supporting the JVM.

The AHA engine maintains an user model which profiles students according to their knowledge, course concepts and GUI preferences. In De Bra's description of the engine [DEB99]:

> "Knowledge is generated by reading pages and taking tests. The (textual or multimedia) content of a page can be adapted by means of *fragment variants*. The (hyper)links are annotated by changing the colour of the link anchor….the colour scheme can be configured by the author and overridden by the user, to choose between *link annotation* and *link hiding*. When desired, link removal can easily be implemented".

AHA pages are generated on-the-fly. HTML fragments contain the learning content, which will form the body of a page, and *html* comment lines, which define prerequisite knowledge and the final document title. The prerequisite knowledge line is used in combination with session tracking and log files to support cognitive structures. It is used during course production to *make* a document list file. The list file is static and defines the complete course structure. The log file is also key to the adaptive link annotation mechanism. When the student first logs into a course he sets existing knowledge levels and presentation preferences. These are then incorporated into the course page when it is generated. JavaScript and cascading style sheets implement it on the client side.

Session tracking is provided by the use of usernames and session codes. This information is incorporated into http request commands. The servlet processing the http request parses the link, updates the log files and incorporates the session data into the hyperlinks of the resultant document. Thus state information is maintained even using the http protocol. Synchronous communication is not catered for although a message board provides for limited group collaboration.

Creating learning material for the AHA engine requires very careful authoring. Fragments can be created using an HTML editor but the processes of course design and content authoring are interdependent. The course structure is static and should be planned out

carefully prior to content being created. The interface is well supported with navigation aids. Their adaptive nature aids rapid movement through a course and reduces disorientation. It neither supports course management tools nor is suitable for integration with other reporting, curriculum or administrative support systems. The adoption of HTML as the information format though permits the incorporation of multimedia content. Interactive testing can be provided using scripts.

5.4.2 Courseware on the fly

In "Courseware On The Fly" Dillon developed an course engine which delivered learning material adapted to user learning requirements. Learning material in the form of *pagelets* was stored in an Oracle database. Lesson pages were generated on-the-fly. As with AHA, JavaScript was used to control presentation on the client side. Now, it is also used to collect and process student input from *html* forms. Dillon's use of a relational database permitted the modelling of a more sophisticated and dynamic structure and the ability to modify course structures with ease. Java applets connected directly with the database using proprietary protocols and translated the results into *html* documents. The size of the applets required to implement the solution was large. This was recognised by Dillon as a matter for further investigation. [DIL98].

Java Script was used to process a user questionnaire (prior knowledge) and establish an initial student profile. This was used to select a course roadmap. Only basic AI was used. JavaScript was also utilised in test pages automating correction and feedback processes. Ease of use was addressed through the interface design. Frames were used to display menus, learning material and navigational icons. However, page sizes were large and required scroll bars. Interaction was not limited to navigation and testing: in the example SQL course, a simulation tool lets students construct SQL statements and test them from their browser. An email service was provided, however for neither synchronous communication nor collaboration.

Neither student login nor integration with faculty, support or curriculum development systems were supported. Nor was it possible to adapt course structures to sophisticated profiles.

5.4.3 Educational Courseware Design for the Web

In Conlan's courseware engine [CON99], the AI used to generated course structure was developed further. Metadata was introduced to define and describe learning components.

Components conformed to a hierarchical structure of course, section, unit and pagelet. Each one was described using five groups of metadata: general, semantic, pedagogical, technical and conditions of use. These groups were mapped to the Dublin Core system of XML metadata. Both textual content and link information relating to the components were included in an Oracle database. The engine was implemented using Java and CORBA. The GUI was multiplatform capable, having been constructed using Java Swing .

Conlan set out to improve on Dillon's application. Like the latter he used previous knowledge as the base line. However, the user also specified a goal. These were used to querying the course structure with the result that a totally personalised, or adapted, road map was generated. This not only supported cognitive structures to a greater degree, it also enabled component reuse with out the need for further human input.

HTML was not used for presentation. Because CORBA is not fully supported by web clients, the selection of Orbix Web as the implementation technology limits the deployment options. Course, resource and user/group management tools were provided. However integration, collaboration, reporting and testing were not. But Conlan's showed that the structured and consistent use of meta- data could be an effective basis for developing dynamic JiT courses.

5.5 Productivity Tools

5.5.1 Microsoft PowerPoint

PowerPoint is arguably the leading integrated presentation tool. It provides a vector graphics work area, a text editor for producing notes and a text based presentation development tool. Slide shows can be delivered using overheads or via a projector. Projector delivery can exploit PowerPoint's support for slide transition animation and embedded multimedia objects. Slide shows can be synchronised and carry soundtracks. Standard delivery requires a Windows platform.

Recent versions of PowerPoint include tools for converting presentations to Web compatible formats. By default, graphics are converted to GIF images, notes to HTML text and the slide structure to hypertext. These components can be combined in many configurations using a wizard and contained within a *html* frameset. A presentation can

thus be delivered on any platform supporting *html* frames. However animation and object linking are lost.

An ActiveX component is also available to present a movie version of the slide show. In this format slide transitions, internal animations and sound effects are maintained. However multimedia content is in its raw form and file sizes are consequently large. However, slide shows can also be converted to NetShow, *asf*, movies. As such graphics are converted to *gif* images but the synchronised sound track is compressed. The entire slide show is defined using a metafile and can be streamed to the client. Real Systems has a tool which can convert the slide show to a cross-platform *rm* movie.

The *html* conversion tools available for PowerPoint make is an obvious initial *html* content creation choice for those familiar with the application.

5.5.2 Macromedia Flash

Macromedia Flash is a vector graphics drawing application. It is used for the creation of interactive multimedia movies and has excellent support for creating animations. Flash movies can be saved in their native format or exported as cross-platform ShockWave Flash (SWF) or animated *gif* files. SWF files are interactive and much smaller that than animated *gif* files. They also resize without distortion. Furthermore, sound can be streamed to a web client. Movies can be embedded in *html* pages and run using browser plugins or ActiveX controls.

The considerable functionality afforded by the Flash application takes time to learn. Thus it is most suited to the needs of the multimedia professional. However, its interface is intuitive and the novice can learn to produce simple movies quite quickly.

5.6 Conference Tools

5.6.1 Microsoft NetMeeting

Microsoft NetMeeting is a video and data conferencing tool. It is included as part of the full Internet Explorer (IE) 5.0 installation. Thus, its cost of ownership is negligible and its availability within an enterprise setting high. High availability can easily lead to wide-scale user familiarity. Conferencing concepts were discussed in chapter 2. NetMeeting ActiveX controls can be embedded within *html* pages . For those with the full IE5.0 installation, video, chat and application sharing can be invoked by simply requesting a

web page. Plug-in support is also available for Netscape on Windows platforms. Thus quality interfaces can be created to meet user requirements.

NetMeeting controls include chat, a shared whiteboard, a video viewer and file transfer as standard. Application sharing turns one host's display and the applications running on it into a shared workspace. Application control can be retained by the host or passed to other participants. More than one application can be shared at once. Unfortunately, NetMeeting supports only one-to-one audio and video conferencing.

5.6.2 CuSeeMe

In addition to the standard data conferencing tools, CuSeeMe (White Pine Software) permits audio and video conferencing for up to 12 hosts. It is, consequently, an appropriate application for implementing discussion groups . It offers multiple-platform deployment, though not yet on Unix. However it can only function as an external application. Thus it does not integrate seamlessly into web pages. Nevertheless, calls to other CuSeeMe clients can be made from within a web page, which makes its use relatively simple.

5.7 CBT Course Analysis

The most sophisticated examples of the application of courseware are to be found in the training rather than educational sector. However, even there, they do not fully address learning. In this section we will examine actual courses produced using technology described earlier in this chapter. It will offer an insight not only into the capabilities and limitations of the applications, but also into how imaginatively technology has been applied.

5.7.1 DDI Personal Interactive Learning Series (PILS)

Development Dimensions International (DDI) is a multinational training and consultancy company which supports the corporate business sector. Certain courses have been developed using Asymetrix software. They can be deployed over an intranet or using a CD-ROM. The PILS series is an example.

The course reviewed, "Targeted Selection", made full use of a wide range of interactive multisensory content. The user interface presented text supported by audio (*wav*) and movie (*avi*) clips. Text was used to summarise the information presented using the other

media. Thus it was concise and easy to read. There was no need for scroll bars. The course had a hierarchical structure. Courses comprised modules. Each module had a specific objective. This was displayed to the students. Learning material was presented as a sequence of pages. Menus were used as the primary format for presenting structure and providing navigation. A statement of the learning objectives was presented at the beginning of each module. Although no summary was provided at the end, an online book was available for background reading.

Supporting material included a course map, a searchable note taking tool, an online manual, a searchable glossary and evaluation tests. Evaluation took the form of multiple choice self test checks, video scenarios and practical assignments. Synchronous network communication and collaborative workspaces were not provided. Nevertheless, students were set assignments for completion in class, under the supervision of a tutor or over the telephone. A basic tracking tool permitted students to bookmark their progress though the course, and a data reporting utility fed back management information

This course was very sophisticated and had addressed most of the criteria listed earlier in the chapter.

5.7.2 Microsoft Research (MSR) 1999 European Briefing

Microsoft Research (MSR) recorded the proceedings of their 1999 European Briefing and distributed the recording on a CD-ROM. Although not a teaching package in the strictest sense, the recording makes a useful example for analysis. It has already been shown that Microsoft has developed some interesting technologies and founded upon a common architecture utilising Windows and COM. Many technologies integrate seamlessly so could form the foundation for building distributed course delivery applications. For the following analysis it is assumed that Windows is an acceptable environment and Internet Explorer the web client.

The CD-ROM package comprised a series of lecture presentations before a live audience, which provided interaction in the form of questions. The speaker stood before the audience and used visual aids in the form of PowerPoint presentations or other audio-visual material. The speaker would move between the lectern and screen. The lecture was recorded using a video camera, which had an operator controlling direction, zoom focus etc. In many regards the format would be representative of a typical academic lecture, albeit without the a video operator.

The technology used was streamed video (*asf*), PowerPoint presentations (*ppt*t) and *html* frames (see PowerPoint above). NetShow could be used for network serving. Users could choose between seeing the original presentation (typical size 40 slides and 2 Mbytes), browsing the *html* presentation (typical aggregate size 2 Mbytes) or watching the lecture. In the latter case, slides could also seen, as the video operator would redirect the focus. However, although the slides could be read, the quality was not as good as that provided by the other formats. In the default view, multimedia technologies were not mixed.

In the lecture given by Anoop Gupta several interactive multimedia research projects were described. One, Flatland, was a tele-presentation system. Though focussed primarily at the educational sector, Flatland could be considered as a flexible architecture for distributed collaborative applications. Clients could view a presentation via a DHTML client. Streamed video was used to stream the presentation in pseudo real time (network latency averaged about 8 seconds) and embedded within an *dhtml* page. Client activity was captured and sent to a collaboration server, which co-ordinated the other page components, thence overcoming the latency problem. Other components included PowerPoint slides, chat and other collaborative tools. Trial feedback suggested favourable student reception.

In comparison with the Flatland system, the MSD briefing package was very thin.

Multi-sensory media were used, but in isolation. Little attention had been paid to the design, hence quality, of the interface. Video and slides could easily have been combined within single *html* page. The absence of this denied the video viewer the ability to follow the lecture flow in textual form, the presentation viewer access to audio narrative and lecturer gesticulation and both access to supporting notes.

On the plus side, creating the content had obviously been quick, simple and low cost. Furthermore the selection of *asf* as the video format permitted effective deployment over the Internet or using a CD-ROM (This method permitted client applications to be shipped with the material. Alternatively, it can be downloaded for free.)

This example demonstrates, that despite the availability of powerful technology and tools, unless they are employed thoughtfully and imaginatively their value will not be realised. In delivering education, training or a presentation the audience should remain the focus for the speaker. In an online context, that begins with a well designed user interface.

5.8 Synthesis

In this section we will review the all the research presented so far and see if it supports the proposition:

> that current technologies can be combined in imaginative ways and with relative ease to satisfy specific learning objectives; technology can assist with the process of defining learning objectives and supporting the provision of education; a unified distributed architecture will be required to support the system and open standards to ensure a consistent level of student access.

On the basis of this analysis, a description of a possible prototype solution will be presented with a view to implementing it for evaluation.

5.8.1 Education

In the examination of teaching theory it was concluded that the role of the teacher is create an environment that supports learning through individual and group activities. Learning objectives apply not only to the knowledge domain; the affective domain sets objectives within a personal and social context. Individual have preferred learning styles, which if supported enhance the learning process. Thus effective learning is not achieved simply by presenting information: learning requires planning, structure, adaptation and variety.

5.8.2 Technology

In our examination of multimedia we saw that a complex variety of formats and standards exist. Multimedia content in its raw form is large and difficult to deliver over a network. Compression and streaming technologies are emerging to ease delivery. Many use proprietary standards. They require specific client applications to present them, and servers to gain full benefit from them. HTML was shown to be an effective text presentation medium and a conduit for delivering more sophisticated content, such as dynamic graphics, applets and the more popular multimedia formats. But, the implementation of *html* has become far from standard, so exploiting it fully requires skill and experience.

Databases were discussed. Object relational databases can organise both complex data structures and complex data types. Databases not only store and structure information, they can also be used to deliver it. Oracle was offered as an example of a DBMS with

direct support for XML. ODBC compliant databases can be used to produce *html* pages on-the-fly with. Access to the database can be via a middleware application linked to a web server. It was suggested that Java servlets provided an effective mechanism for doing so.

5.8.3 Delivering Education

In the chapter on delivering education it was argued that using structure and standard methods in the design process helps with defining learning objectives and the effective use of multi-sensory activities.. This can also simplify the implementation of adaptive presentation and provide the opportunity to share content and course components. It may also lead to better quality control and an integration of course development, delivery and administration systems. The inter-relations required to support such a system are complex.

Two forms of adaptivity were considered. Navigational adaptivity modifies the appearance of page links in accordance with a user profile, so helping the student find an appropriate path through the learning material. Structural adaptivity reconfigures the course makeup to the needs of the student. The distinction is analogous to a book's contents page which helps the reader find the desired text and the index which offers up a selection of texts based on metadata. Open and dynamic information domains require structural adaptivity, supported by intelligence and an agreed vocabulary of metadata.

Delivering information to the user should be consistent. It was argued that *html* and web clients provide familiarity and relative consistency. The ability to incorporate a wide variety of media and third party application interfaces reinforced its attractiveness. However, incorporating such media under conditions of widescale delivery, is far from straight forward. There is a requirement for experienced multimedia resources to advise and assist with its implementation.

5.8.4 The State of the Art

At the beginning of this chapter criteria were presented which could be used for evaluating educational software. We then evaluated several courseware applications addressing course development and course management issues. The results were summarised in figure 5.1. and showed the sufficient functionality in order to develop,

deliver and administer courses and to create the components necessary to implement individual and collaborative learning.

Course structures were generally hierarchical with content created externally to satisfy the requirements of a particular learning components. Components combined different media to provide variety and satisfy different learning styles. Movement through the course structure was afforded using a combination of menus, buttons and hyperlinks. Course development applications used a combination of graphical and question-based wizard techniques. Both set out to construct a hierarchy and to define components in a standardised manner. Course management applications incorporated student and class profiles. Student tracking was required and achieved in several ways. All provided access to material through a server supporting registration and login utilities.

Some of the applications reviewed were platform specific or used proprietary media formats. Of these, many provided the facility to convert content to *html*. Specialist authoring tools were presented which produce multimedia content in a format appropriate to the heterogeneous nature of the Internet. They tended to have user-friendly point-and-click interfaces, but offered much functionality besides. HTML editors could generate *html* scripting from drag-and-drop operations. Some offered support for *xml*.

Most integrated courseware applications incorporated email and discussion groups. Few included conferencing as standard. Conferencing tools were shown to be readily available as stand alone applications. Some were free; others low cost; most could be accessed from *html* pages, some could have their controls embedded. Not all conference applications were cross platform or supported many-to-many video. The ability to share applications offered the teacher the possibility to demonstrate on line in real time.

5.8.5 Conclusion

Educational courses should comprise learning components which conform to a planned curriculum. Technology can be used to support the development process and to implement the delivery of learning material and activities. Despite the availability of technology, not all course designers exploit it imaginatively or fully. Existing applications already possess the tools necessary to do so. High quality interfaces require careful design and good tools support. A single distributed architecture can be used to bind the entire system, while still providing open access and flexibility in how it is used.

5.9 Statement of Requirement

The remainder of this document will concern the development of an distributed architecture to support the provision of educational for deployment over a TCP/IP network. The architecture should support the structured development of and open access to course material and flexibility in how the teacher may uses multi-sensory media to support learning activities. Off-the-shelf productivity tools should be used where appropriate. The following should be supported:

- Course design and development
- Creation and integration of interactive multimedia content
- Collaborative activities
- Course production and delivery on-the-fly
- Adaptivity and course administration

Chapter 6 A Unified System Architecture

6.1 Introduction

In the last chapter it was concluded that existing technology could be used to implement a comprehensive integrated course development, delivery and administration application. Structured learning material comprising multimedia components could be created using commercial tools and adapted to support individual learning objectives and group learning tasks. Courses should be produced on the fly and delivered to students via standard web interfaces.. In chapter 1, it was proposed that a prototype be developed and used to create a course owing functions:

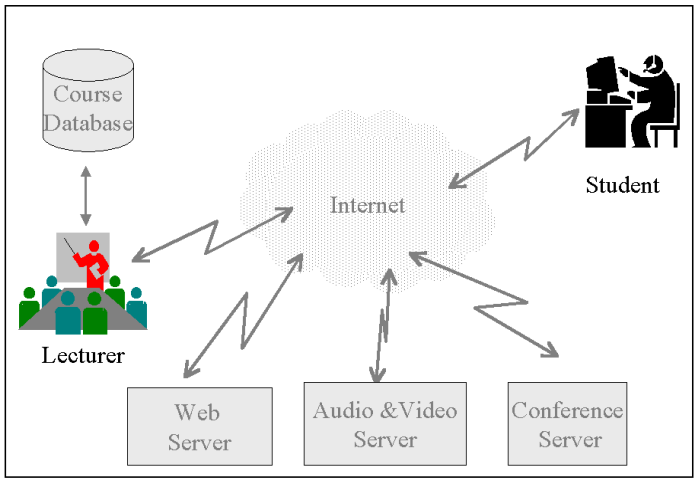

Figure 6.1 A Unified Distributed Courseware Architecture

- Course design and development
- Creation and integration of interactive multimedia content
- Collaborative activities
- Course production and delivery on-the-fly
- Adaptivity and course administration

Figure 6.1 illustrates a unified system architecture on which to build the application. In this chapter the above functions will be addressed in turn. Technologies satisfying the requirements and conforming to the architecture will be identified. Issues relating to their implementation from a technical perspective will also be presented.

6.2 Course Design and Development

A design tool should assist the course developer with building the course structure and setting learning component descriptions in accordance with its curriculum. Courses will be hierarchical in structure. Lessons will comprise elements, each of which can be accessed by its URL. The developer will make use of the tool through a GUI. It should be simple to use. The tool should also be able to contribute to the management process in supporting shared access and management reports. A relational database will be used for implementing it.

6.2.1 DBMS Selection

Oracle supports binary data, user-defined types and XML and is well established in the corporate arena. However, its user interface is not as friendly as that provided with Microsoft Access, the latter will be used for the implementation.

The Access interface uses point and click controls to assist in developing tables and queries. Using its query generation wizard, SQL query strings can be developed and tested quickly. The Forms and Reports views are particularly useful and easy to use. Forms are designed for screen input. Nested forms allow data from several related tables to be updated on a single screen. The Access report wizard automates the process of report generation.

Microsoft Access includes an ODBC Manager as part of the Windows installation. It has support for replication and synchronisation as well as a web interaction tool for exporting queries and tables in *html* (*asp*).

6.2.2 Database Design

The course structure will be stored in tables. Inter-table relationships will be specified. A simple tree structure is recommended where possible. However, it is possible that a single learning component could be part of several courses. Thus, it is desirable that learning components can be reused. Simple trees can support one-to-many relationships. Reuse requires man-to-many relationships, so intermediary tables will included between each level to support it.

The following tables will be created to support a 3-level hierarchical course structure:

- CourseDetails - course_ID, title, description, director, etc
- ModuleDetails - module_ID, title, description, tutor, etc.
- PageDetails - page_ID, title, description, author etc
- ElementDetails - element_ID, description, creator, etc
- CourseModules (intermediary table) - course_ID, module_ID, module sequence #, etc
- ModulePages (intermediary table) - module_ID etc
- PageElements (intermediary table) - element_ID etc
- PageCategories - (supporting table) page formatting information
- ElementTypes - (supporting table) element formatting information used for frame delivery
- URLDetails - (supporting table) to assist with managing URL base addresses

Note the use of supporting tables to speed input and reduce keyboard errors and personnel data. The hierarchy director, tutor, author and creator is a possible staff structure for realising the curriculum. DBMS flexibility makes the process of adding to a tables straight forward.

Additional tables will be added to support administrative and profiling functions and will be discussed later in this chapter.

6.2.3 Database GUI Access

The database GUI will be the primary point of contact for the course designer and will be used to develop the course structure. Access forms take table or query

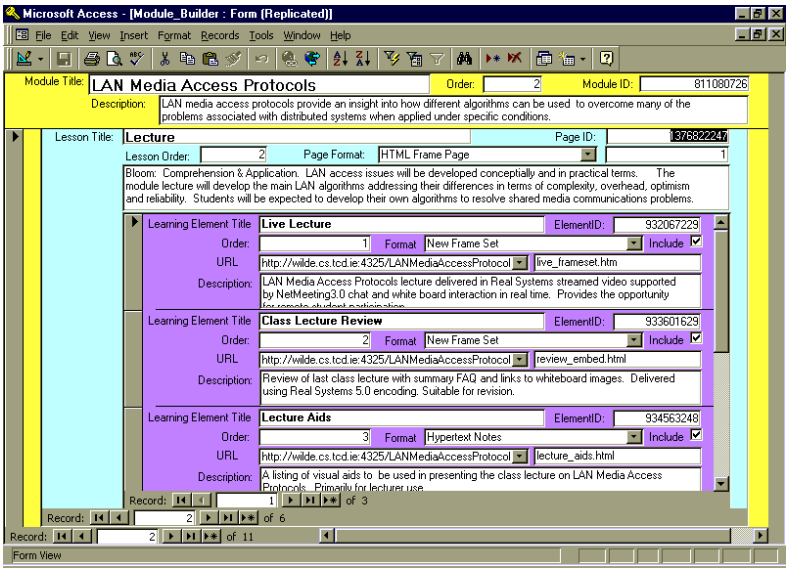

Figure 6.2 Module form showing 2-level sub-form nesting

entries and display them in a more user friendly manner. Additional material not included in the tables, for example instructions, can be included on the form. Browsing controls aid navigation between entries.

Access permits the use of up to two levels of nested sub-forms. (figure 6.2). Parent and child fields can be linked, facilitating a grouping of entries within the subform. Sub forms can be browsed within their parent. Similarly for sub-subforms. Subforms will be used extensively in the GUI designs.

6.3 Creation and Integration of Interactive Multimedia Content

Multimedia content will be restricted to the following formats:

- Text - *html, txt, doc* (Word 95)
- Images - *gif, bmp, jpg*
- Audio - *wav, ra, mpeg*
- Video - *avi, mov, mpeg, rm, asf*
- Animation files - *swf, fla, avi, mov*

HTML will be the primary presentation format. Thus an *html* editing tool will be required. Frames will be used to aid visibility. Other multimedia content will be embedded within a web page where possible. In addition to an *html* editor, video inline editing, graphics, conferencing, animation and encoding applications will be required. These will mainly be off-the-shelf. For reasons explained below, the preferred video format will be *rm*; the animation file format will be *swf*. Both are supported by cross platform plug-in applications.

For the prototype implementation only, Internet Explorer 5.0 will be the preferred browser. This contradicts the principles of openness and cross-platform computability. However, the use of ActiveX controls in Web documents should be investigated. Thus, ActiveX controls will be used where possible in parallel with Netscape plug-ins where available.

6.3.1 HTML Editor

The *html* editor should provide a user friendly point and click interface and support dynamic *html* and *xml* if possible. Of particular importance are its tools for simplifying

scripting process. Because *html* documents will also be used for containing other MIME-types, support for plug-ins and ActiveX should be provided. Macromedia Dreamweaver will be used for the pilot implementation.

6.3.2 Streamed Audio and Video

Although the pilot will run on a Windows platform, cross platform content is preferred. Real System's media (version 5) meets the requirement and will be the preferred audio-visual format. Real System media are also supported by the JMF, which adds weight to the selection.

Streamed Video Server

Real Server is required for live delivery and represents the most scaleable solution for delivering static compressed audio and video. For performance reasons, the server will be installed on its own platform. It serves content using the PNM, RTSP and HTTP protocols from ports 7070, 554 and 8086 respectively and receives it through ports 4040 and 5050 (older formats such as 5.0).

Video Capture and Encoding

Real Encoder 5.0 will convert live or other audio visual media to Real Media 5.0 format. Although not the latest format, this format can be delivered over Unix using RealPlayer 5.0. The encoder will be installed on the same workstation as will be used for capturing the source. Two forms of raw video will be used: live and video tape.

Live Capture. A simple video camera will be required for video conferencing and live compressed video presentations. A low cost USB Web camera will be connected to a laptop computer running Windows 98. Camera support for Windows NT and Unix is currently very limited. The camera output will be encoded then streamed to the media server as outlined above. Media clients will connect to the server using the PNM protocol. The encoder will also save a local copy of the media for subsequent review.

Video Tape. Video clips will be also be captured from tape, digitised and converted to *rm* 5.0. Three streaming formats will be required: 28kbps (Unix), 56kbs (general) and 100kbps (LAN). Technical assistance will be needed. The steps involved in capturing, encoding and converting the clip are outlined in chapter 7. Audio-Visual and Media Services, Trinity College have offered to help with the trial.

Compressed Video Client

Real Player will be the default streamed video client. Content will be accessible from web pages. Presentation options will include both embedded and via an external player. Specimen script for doing so is provided in the annex . Web clients cannot access it directly and require meta file support as outlined below:

- Direct player access. An external player can access the content using the appropriate call - protocol (RTSP or PNM): URL. e.g. - pnm://my_server/my_video.rm The call may refer to the actual file or to a SMI metafile, providing several play options - e.g. rtsp://my_server/my_options.smi .
- RAM file access. A *ram* file can be stored on the web server and can make the call by proxy. It can be treated as a normal hyperlink e.g.- http://my_webserver/my_video.ram
- RPM file access. Embedded and ActiveX use a *rpm* file as the proxy. It contains the came information as the *ram* file. When embedding the player control the address of the *rpm* file should be specified within the script e.g source="my_video.rpm"

6.3.3 Animation, Graphics and Other Static Multimedia Content

Most static multimedia content will be served from a Web server. Standard commercial authoring tools will be required to create the material. Being cross-platform compatible, Shockwave Flash will be the preferred format for animations. Dreamweaver templates will be used for embedding embed content in web pages using plug-in and ActiveX controls.

6.4 Collaborative Activities

Collaborative activities will be supported in two components. The lecture will be broadcast live in compressed video format with collaboration and feedback provided using NetMeeting data conferencing. All controls will be embedded within a web page on the student's browser. Group discussions will be facilitated using CuSeeMe as a external application. Both CuSeeMe and NetMeeting installations include client and server processes. They will be installed on Windows hosts as standalone applications.

6.4.1 NetMeeting

A conference server will be invoked on a host via the "host meeting" command. Other hosts can contact the server, by calling the "host" machine. The server can accept calls automatically.

Participants can invoke their clients by starting the application manually or by requesting the web page containing the embedded controls. The embedded option makes the interface less cluttered, because the page author can include only those controls which are necessary. In the configuration used in the prototype, the video display was not included. The meeting "host" can specify the rules for what will be shared and how control will be managed.

Information contained in the whiteboard and chat windows can be saved on completion of the lecture. Specimen JavaScript code segments are provided in the annex.

6.4.2 CuSeeMe

CuSeeMe provides two options for hosting meetings: group and multicast conferences. In the latter case, audio and video are multicast to all participants in simplex mode. Group mode will be used for the group discussions. Figure 6.3 shows a group conference with five participants. The meeting "host" will invoke a conference server using the "Create Multicast Conference" command. Several configurations are possible and a

Figure 6.3 CuSeeMe group conferencing

wizard is provided to assist. Address details must be allocated to the server e.g.

- Conference ID: 31355;
- Audio and Control: 224.2.2.2:27648
- Video: 224.2.1.1: 57648.

The use of multiple IP addresses can present many difficulties for firewalls and web proxy servers. Other hosts can join a conference by calling using the multicast tool in the client 's default window. Alternatively, conference details can be specified in a meta file with the extension ".cu", and located on the web server. This method will be used for the prototype.

Initially, both web server and client must be configured. The web server must have "application/x-cuseeme cu csm" included in its mime.type file. The web client must associate it with the CuSeeMe application file. A metafile must is created and located on the server, e.g. as shown in file "pc535_conf.cu" below. A link to the file can be included in a web page. When a web client downloads the link, CuSeeMe will

connect join the conference automatically.

Pc535_conf.cu:

pc535.cs.tcd.ie134.226.38.4 0

[Settings] Max Windows=4

[Flow Control] MaxCap=56000 MinCap=9600 UseFlowControl=Yes

[Connect Options] IWillSendVideo=Yes IWillRecvVideo=Yes IWillSendAudio=Yes IWillRecvAudio=Yes

6.5 Course Production and Delivery On-the-fly

Production on-the-fly permits dynamic course structuring. A course manager (middleware application) will generate *html* pages reflecting the course structure from link and metadata stored in the relational database. These pages will reference course components stored on other servers. The database will be the same one as was used for course development. Live Software's JRun is a Java server which can run as a servlet extension to certain other web servers. Compatible servers include Apache, IIS and Netscape

Enterprise Server. It will be used to invoke servlets within the course manager. Further details are provided below.

A Web client will request course roadmap from the main web server. It will pass the request to the course manager on the same host. The course manager will use JDBC to connect to an ODBC manager which will pass a SQL query to the database. The appropriate structure will be passed back down the line and converted to *html* before being returned to the web client.

6.5.1 Integrating the Servlet Extension

It was decided to use JRun in conjunction with Apache version 1.3.6. Both servers will be installed on a Windows platform giving JRun direct access to the ODBC manager. JRun can be administered using an applet-based GUI from a web client

An Apache alias, "servlets", will be used to direct servlet requests to JRun. The default JRun servlet directory is "servlets". As recommended by Moss [MOS98], Java packaging will be employed. Course specific servlets will be grouped in directory tree called Courses; helper servlets written by Moss will reside in another, "javaservlets".

6.5.2 Database Web Access using Servlets and JDBC

Using an Java server, servlets and JDBC in a middleware configuration supports cross platform deployment. The mechanism for querying the courses database is outlined below. In brief, a Web client will request a servlet from the extension service using html. The servlet will open a connection to the database using the jdbc-odbc bridge, pass an SQL statement, receive a result set, parse it, then generate an html page on-the fly. The web server will return the page to the web client for presentation. Further details are provided below.

Servlet invocation

The book Java Servlets [MOS98] provided much guidance on how to employ servlets for database connectivity. Its comprehensive selection of sample programs were used as the basis for the servlets used in this dissertation.

A servlet can be invoked from a form or a simple hyperlink. In this implementation all servlets will be accessed using the "servlets" alias and their full class name. For example:

- <a href="http://my_myserver/servlets/courses.Courselist">

- <a href="http://my_server/servlet/courses.ModuleList?key=1">

The first example simply requests information and is implemented using the servlet's doGet() method. The second also submits search criteria: key=1. Data can be submitted using either post or get actions, according to normal data size and security criteria. They are implemented using the servlet's doGet() or doPost() methods, in this case the former. The data submitted with servlet requested is included within the HTTP request. It can be extracted at the server and incorporated into a SQL statement. The servlet is returned a result set, which it parses, formats, then returns to client as an *html* document.

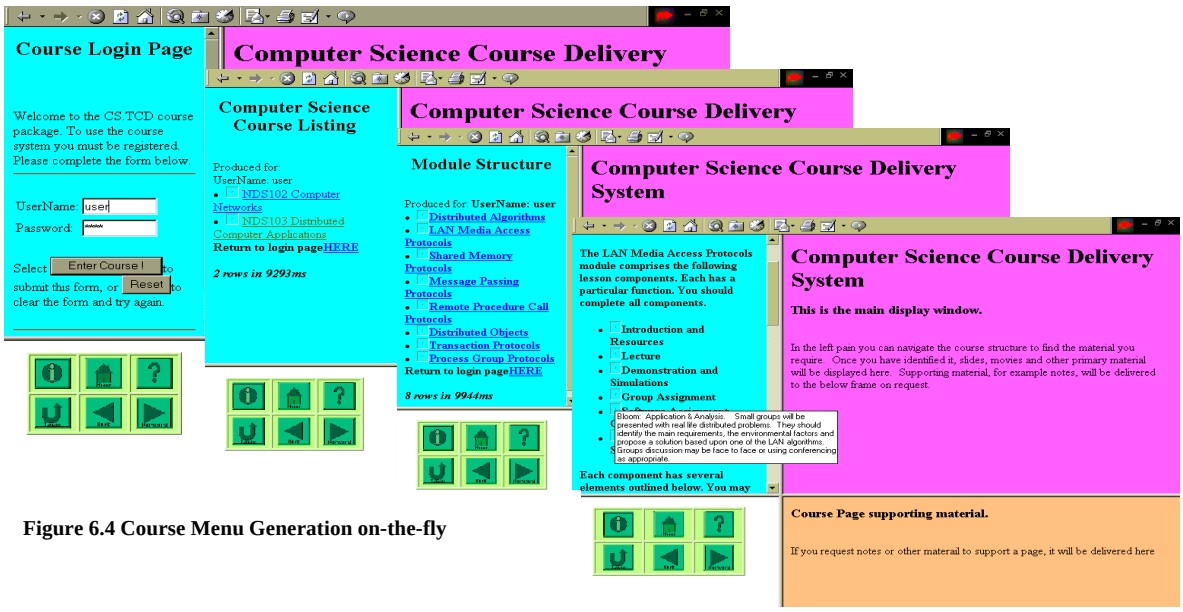

Figure 6.4 Course Menu Generation on-the-fly

Figure 6.4 shows a sequence of on-the-fly frames which present a hierarchical learning structure down to lesson component level. The menu contents, including the look ahead box detail, and are initiated by a hypertext request as described above.

Database connectivity

Connection overhead can be reduced by establishing a connection pool. This is possible due to servlet persistency. The ConnectionPool class (Moss, unchanged) will

be used to set up a series of connections to the database on initialisation in accordance with a configuration file shown below.

```
#ConnectionPool.cfg
        JDBCDriver=sun.jdbc.odbc.JdbcOdbcDriver
        JDBCConnectionURL=jdbc:odbc:CourseMngr
        User=my_user
        Password=my_password
        ConnectionPoolSize=5
        ConnectionPoolMax=100
        ConnectionUseCount=5
        ConnectionTimeout = 2
```

The JDBC driver shown is the standard jdbc-odbc bridge. As stated above, the database format will be Microsoft Access. Windows includes an ODBC manager, accessible via the Control Panel. This will be used to set up the ODBC driver (Driver name: CourseMngr, Username: User and Password: password). The remaining parameters in the configuration file refer to the connection pool.

The connection pool will be set up when the servlet is first initialised. Because servlets are persistent, the connection pool will remain resident in memory and available for other servlets.

Database Query

The process of querying a database is relatively simple. In brief, a SQL query will be formulated and converted to a JDBC statement. A connection to the database will be seized and the query executed. The database will return a result set object. On completion of the query, the connection will be released and the result set processed.

Ease of use is an important design criteria and one of the primary reasons for selecting Access as the DBMS for the prototype. Its point and click query tools simplify the SQL statement design process. Access will be used to generate the main query tables (see below), which servlets will query in turn using simpler SQL statements.

Figure 6.5 illustrates of the composition of the Courses database. The lines linking the tables define the relationships between linked fields. The figure illustrates how curriculum development, course management and administrative functions can be

integrated. The table CourseDetails is linked to CourseModules (structure) People (development) and Classes (administration). SQL represent the relationship using the: "INNERJOIN… ON….= …..ON" construct.

Note, INNERJOINs are nested and reference is made to data items from 6 tables. Access made light work for of constructing this rather complex statement.

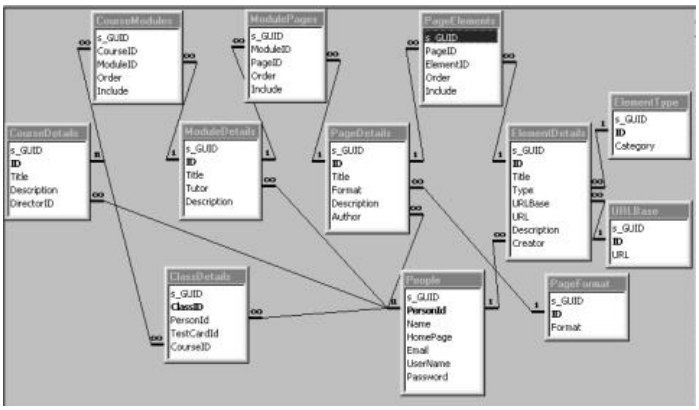

Figure 6.5 Course Database Structure

SELECT URLBase.URL, ElementDetails.URL, CourseDetails.Title, ModuleDetails.Title, PageDetails.Title, ElementType.Category, ElementDetails.Title

FROM ElementType INNER JOIN (URLBase INNER JOIN ((PageDetails INNER JOIN ((ModuleDetails INNER JOIN (CourseDetails INNER JOIN CourseModules ON CourseDetails.ID = CourseModules.CourseID) ON ModuleDetails.ID = CourseModules.ModuleID) INNER JOIN ModulePages ON ModuleDetails.ID = ModulePages.ModuleID) ON PageDetails.ID = ModulePages.PageID) INNER JOIN (ElementDetails INNER JOIN PageElements ON ElementDetails.ID = PageElements.ElementID) ON PageDetails.ID = PageElements.PageID) ON URLBase.ID = ElementDetails.URLBase) ON ElementType.ID = ElementDetails.Type

WHERE (((CourseModules.Include)=Yes) AND ((ModulePages.Include)=Yes) AND ((PageElements.Include)=Yes))

ORDER BY CourseModules.Order, ModulePages.Order;

The course manager will then be able to make a simple one-line " SELECT * FROM query WHERE" request on the above, which is tidier, easier to do and test and faster to process, than hard coding the full SQL statement into the servlet.

6.5.3 Serving HTML-Compliant Content.

An Apache web server will be required to deliver textual, graphics and other non-streamed content. To improve performance, the content web server will be on a separate host from the course manager. Content will presented to the student using a Web client.

WWW Clients

It was not possible to produce a simple cross-platform solution to the delivery of all content using a single web client. Standards issues were discussed earlier in this document. ActiveX controls currently offer the most seamless solution to supporting embedded applications within web pages. Java, and JMF in particular, is considered a better long term alternative.

Internet Explorer 5.0 will be used as the default Web client. Scripting and plug-ins will be incorporated where possible to permit presentation on Netscape and its Unix client as appropriate.

6.6 Adaptivity and Course Administration

Simple structural adaptation will be applied. Only class members will be given access to the course material. Students will be required to register with the system and login to the site. The process was illustrated in figure 6.4 above. Once logged in, student state information will be passed between the web client and course manager. Because HTTP is not a stateful protocol, a user identifier will be built into page links (figure 6.4) and included within HTTP requests. Thus, state can be maintained and user tracking incorporated.

User profiles will be stored in the course database. This very basic implementation will use two data tables.

- People. This will maintain a record for all registered users. A unique user identifier (UserName) will be created.
- ClassList. This intermediary table linking People and CourseDetails will list those users who have access to a particular course.

For the prototype, adaptivity will only implemented at the level of course access. However, by maintaining user state throughout all levels, sophisticated adaptivity and analytical student tracking will be possible.

The two tables listed above will also form the basis for a course administration utility. Students will be able to register with the system from a Web page; they will be allocated to courses by administrative staff using an Access form. The "People" table will also be extended to include personal and other administrative details, such as email addresses and

computer passwords. The people table will form the basis of a personnel record system. It will be used for identifying course directors, module tutors, page authors and element content creators. In so doing, it will bind together the curriculum development, course administration, content creation and course delivery systems.

6.7 Summary

In this chapter an unified architecture was outlined which will be used to implement a course development, delivery and administration system. Its components are shown in figure 6.6. The centre of the system is a Microsoft Access relational database. Designers will create course structures using Access forms. Course administrators will be able to register students and allocate them to specific courses. Adaptivity will be used to tailor the course content made available to the students and to restrict access to those courses for which they were registered.

A middleware course manager application will provide access to the course database. It will use servlets, a jdbc-odbc bridge and SQL. Client access to the application will be via a web server using a servlets extension module, JRun.

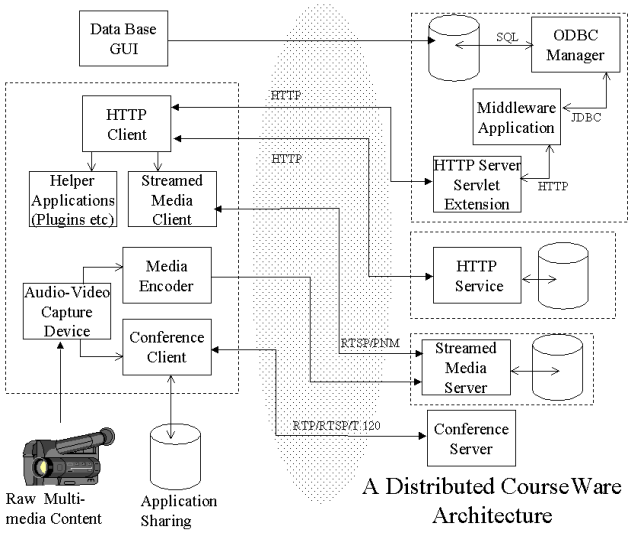

Figure 6.6 Technology Distribution within the Unified Courseware Architecture

The course manager will serve structure dynamically as *html* pages. These will link to the learning components residing on another Web server. User visibility and navigation will be aided by the use of menus in *html* frames.

A series of off-the-shelf multimedia tools would be used for developing content. Content formats should be cross-platform compliant, compatible with *html* embedding and preferably supported by ActiveX controls. IE 5.0 would be the default Web client.

Dedicated media servers will stream compressed audio and video content. Video will be encoded from both live sources and tape. Conference applications will be installed on students and lecturer hosts. Data and video conferencing will be invoked form Web

pages. Meta files residing on Web servers are required by many media servers to provide access to their content from a Web page.

The use of a relational database at the centre of the course production system provides extensibility and the means for integrating it into a larger educational support system. A user profile will be constructed, stored in the database and used to filter page content. The example page sequence showed how both the content and appearance of the pages were linked to the user's UserName. It also showed how learning style and objectives could be incorporated. The look-ahead box described the next page in terms of Bloom and its type. In the prototype, the student will choose suitable material; under roll-out conditions that could be handled by AI.

Chapter 7 Implementation of the Prototype Course

7.1 Introduction

In the last chapter a unified architecture was described capable of supporting an integrated course development, delivery and administration system. Specific technologies were identified to implement the main components and off-the-shelf tools to provide a diverse palette of cross-platform interactive multimedia and collaborative learning activities.

This chapter will describe how the tools were used to produce a prototype hypermedia course module. The module title is "LAN Media Access Protocols", a component from a Distributed Systems Applications course.

The process of developing a lesson structure to support the curriculum using the database GUI will be outlined. It will be shown that information can entered simply and efficiently and that the open access which a DBMS provides permits data sharing across functional lines. Satisfying lesson objectives and the needs of students will considered as the level of granularity reduces. Online lesson formats will be considered, both from the teachers perspective and from that of the student. The hypermedia engine will be presented and a description of how it transforms the curriculum structure and disparate learning activities into a unified course application. Course administration will be addressed. It will be shown how the flexibility inherent in the architecture and selection of technologies simplifies the bolting on of a course administration tool. Furthermore, the facility to query the system based on XML metadata

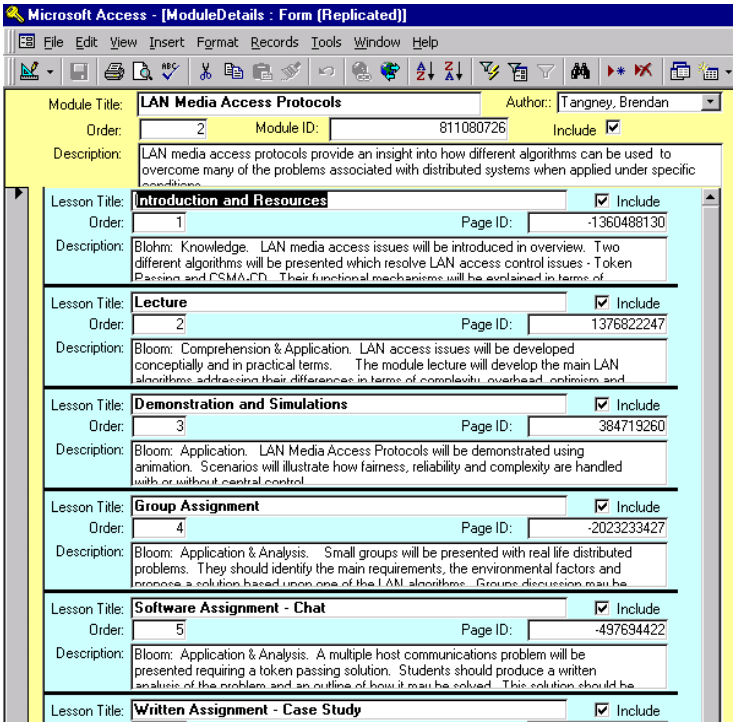

Figure 7. 1 0-2Course Structure Development Tool - Module Form Interface

will be added to demonstrate the extensibility of the system. Finally, web page design will be considered. Interface issues will be addressed with regard to controlling the presentation of static interactive content and dynamic collaborative work spaces.

7.2 Curriculum Development

Curriculum development was taken to be the responsibility of a course director, who identified the requirement for modules and set their objectives. An Access form was used to define modules in accordance with the overall course description. Module details were specified in a sub-form, e.g. the sequence order and the name of the tutor, responsible for its implementation.

The tutor takes the module description specified by the course director and develops a appropriate lesson structure. The three-tier structure implemented in the prototype (Course-Module-Lesson) allotted a learning objective to each lesson in accordance with Bloom's Taxonomy. The Module form, shown at figure 7.1, provided a simple workspace for developing a logical lesson structure. It shows the director's course description and the tutor's lesson objectives. In this scenario, all students, whether campus-based or remote were to undertake every lesson.

7.3 Lesson Planning

At the level of granularity of the lesson, learning style and other student specific criteria were introduced. The database provided a framework to address these issues, without imposing a policy onto how they should be implemented. It also provided a mechanism for introducing *html* page templates and allocating pages to frames automatically.

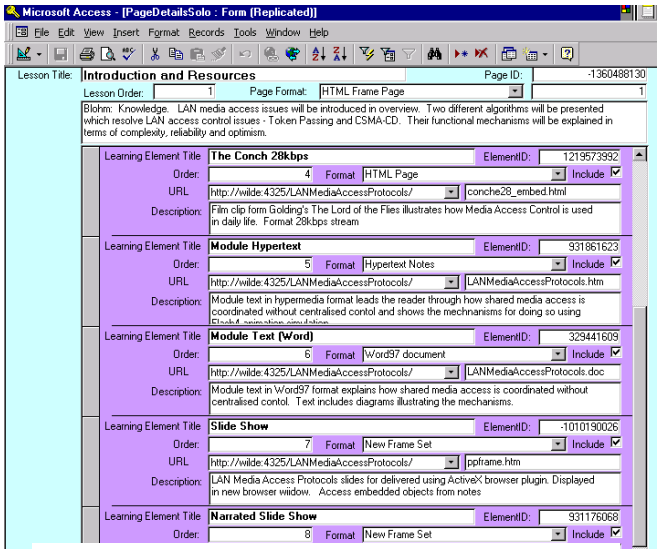

Figure 7.2 Lesson Planning Form Interface - Resource Definition and Mapping Tools

Figure 7.2 shows the PageDetails form relating to the "Introduction and Resources" page. This form is analogous to a lesson plan which may be used for classroom lectures. The main flow is presented together with a list of the supporting aids to enhance the quality of the lesson. In this particular case, Felder's learning

styles were incorporated by providing multimedia components. The form provided the tutor with a work area for selecting content from the rich palette of multi-sensory tools at his disposal. The linking of a component to an URL freed the tutor from the need to author the content at this stage. Nevertheless, the inclusion of template information ("Format") provided and indication of how the tutor would finally like the component presented to the student.

Network considerations were also brought to bear in the selection of learning components. A film clip was offered in three formats: a full screen *avi* version was suitable for classroom projector delivery; a 100kbps streamed *rm* version for LAN online campus students and a 28kbps *rm* version for remote and Unix students. Similarly, background reading was offered in text for those wanting hard copy, or hypertext with embedded *swf* animations for those preferring screen delivery. Slide show presentations were offered with or without a narrative.

The use of the PageDetails form provides much of the functionality afforded by library tools in commercial courseware applications.

7.4 The Course Manager Hypermedia Engine

The purpose of the course manager is to receive a request for course material from a student, to generate an adaptive roadmap and to serve the appropriate content in a format acceptable to the user. The engine carried out three operations.

- A home page was presented as a frame set with information explaining the page format and the mechanisms which the student would meet while progressing through the course. A login form had to be completed successfully to advance further.

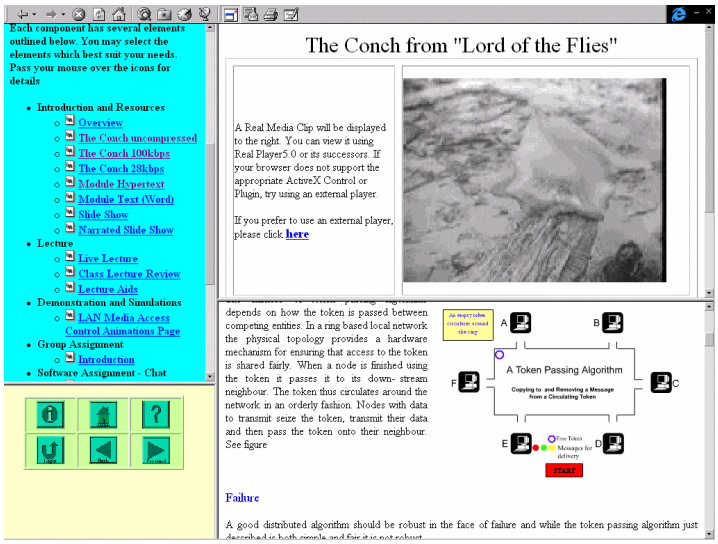

Figure 7.3 Student View Showing Menu, Navigation Buttons and Concurrent Presentation of Multisensory Learning Components

- Structural information was presented in the form of navigation menu. It included the course title and a description of the component. Look ahead (Alt image) boxes were used to minimise clutter on screen. Only material from courses on which a student was registered was provided. The lowest level menu (see figure 7.3) provided a full listing for the module to the level of lesson component.
- Lesson components requested from the menu were directed to the appropriate frame. Thus the tutor had the facility to offer several different activities on a single screen one complimenting the other. Where the default frame set was in appropriate for the activity, different ones were launched in new windows. Activities could be presented in the best format without losing the opportunity for the student to recover his steps.

The screen capture shown at figure 7.3 illustrates the default frameset used to present the module menu (top left), the module notes with supporting embedded interactive animations (bottom right) and a streamed video clip (top right) referenced from the text. The frame panes were resizable, providing the facility for students to adapt the presentation to suit the specific content and their own display settings. Content was developed on a display set to 600x800 pixel resolution.

The hypermedia engine used Java servlets to manage the student login process and to generate the adaptive course structure and link information. The mechanics of using servlets to access a database was covered in the last chapter and its implementation for the prototype was remarkably simple. Only three servlet classes had to be written:

- Courses.CourseList. Generated a menu of courses for which the student was registered together with a brief description of each course in a look ahead box.
- Courses.ModuleList. Generated a menu of the modules included in the course, and their descriptions as above.
- Courses.ModulePageTree. Presented a menu of the lessons comprising the module together with descriptions, the lesson components and URL links targeted to the appropriate frame panel.

Student state information in the form of the login UserName was passed from page to page, thus permitting adaptively to the level of lesson module presentation.

7.5 Course Administration

Basic level course administration was incorporated to demonstrate the extensibility offered by the RDBMS centred architecture.

Student registration was provided through an Access form. An administrator could add an individual to the People table and input personal and other administrative details. Alternatively, a potential student could register with the system remotely via an HTML form. This would be processed by a servlet, Courses.Registration, and the individual would be provided with a default skeleton entry in the People table. A table, ClassList, accessible via an Access form, was used to map students to courses.

The database structure could have been extended further or integrated with other parts of an information system had a greater degree of administrative support been required.

7.6 Support for XML Metadata Queries

The extensibility inherent in the architecture offered the potential to add support for finding learning components based upon XML metadata. A table was added to the database, xml_ims_conlan. This contained metadata in accordance with Conlan's modification of the IMS standard. Mappings were provided to it from the CourseDetails, ModuleDetails, PageDetails and ElementDetails tables. Thus learning components at each level of granularity could be defined in terms of XML resource definitions. Data entry was provided through an Access form, e.g. Lesson_IMS. Courses structures could thus be created based upon metadata alone. This was shown by Conlan. Support for it in the prototype implementation was provided through the database GUI tool. A form, e.g. Lesson_IMS, was used for inputting the IMS metadata. The Access Query tool was used to generate course structures on demand.

7.7 Lesson Component Development

Lesson components were authored using the Dreamweaver *html* editor and specialist content development tools. Most of the components contained static content, that is once created they would not change. This gave the tutor much freedom with regard to a page's final appearance. Lessons based on student participation presented special problems. We will now address some of these issues.

7.7.1 Static Content

The frameset shown in figure 7.3 addressed a particular scenario. The tutor wanted to present all the facts relating to the module in a logical sequence. Text was identified as the most appropriate format. However, text alone would not satisfy active, intuitive, visual and global learners. An interactive animation, however, could support active and global learning styles. The video clip, at first sight unrelated to the module, addressed the requirements of global and visual learners. The tools support offered the tutor a palette of possible multi-sensory media to use; *html*, the canvas in which to employ them and the frameset, a controlled environment in which to combine them imaginatively.

Figure 7.3 shows pages containing embedded *swf* and *rm* files. RealProducer, the successor to RealEncoder5.0, was used to generate both a page template for embedding the video file and the appropriate metafiles to allow files to be played using plug-in, ActiveX or external players. The page template (code listed at Annex C) was then modified using Dreamweaver to add lesson information. Dreamweaver now has its own tools to do the same task and was also used to embed the Flash animation. Writing the code to handle plug-ins and ActiveX controls was considered too complicated to do by hand, but the tools made the task quite simple.

Microsoft PowerPoint was the simplest tool to use, because its *html* conversion tools were completely built in. The final package of *html* pages produced was accessible through a single web page, index.htm. A mapping to it was entered in the database. The conversion process, slide structures and presentation appearance were consistent for any selected template. The format used in the prototype implementation is shown in figure 7.4. The slides were presented in their own frameset together with notes and collapsible menus. Notice that the animation used in the scenario

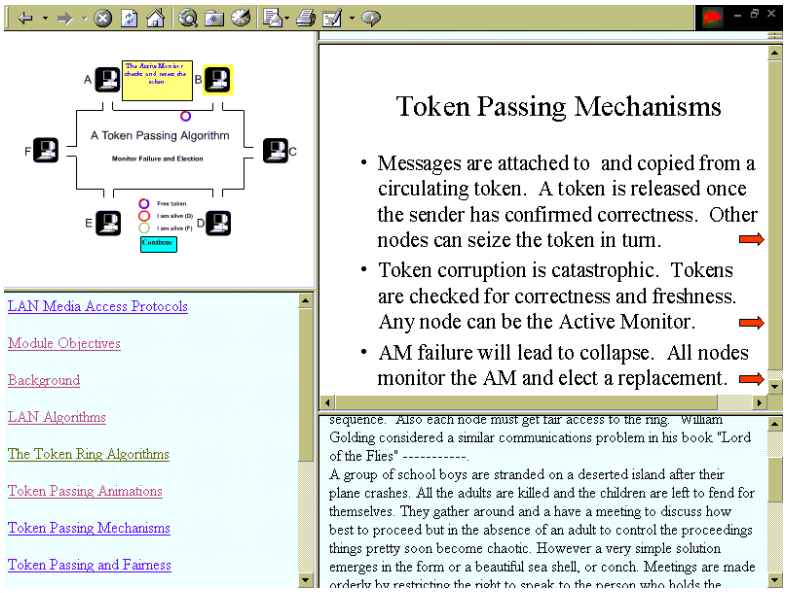

Figure 7.4 A Powerpoint Presentation converted to HTML

depicted by figure 7.3 has been reused in this scenario. The NetShow tool was also used to convert a slide show containing an embedded narrative to a streamed synchronised audio visual *asf* movie.

Flash 4 was used to produce the animation. It is a powerful tool, but takes some time to learn. It is questionable whether most teachers will have the time to do so.

7.7.2 Dynamic Content

Dynamic content is required to support particular scenarios. This could be a discussion group, a live lecture or a demonstration. In the implementation, a virtual classroom was created as a vehicle for extending the physical classroom. On campus students would have the tutor at the front of the class with supplementary material presented on a whiteboard and using PowerPoint slides. Students would be expected to interrupt and ask questions. It would desirable to archive the lecture for review purposes. With this in mind, it was decided to stream the lecture and to provide synchronous data conferencing.

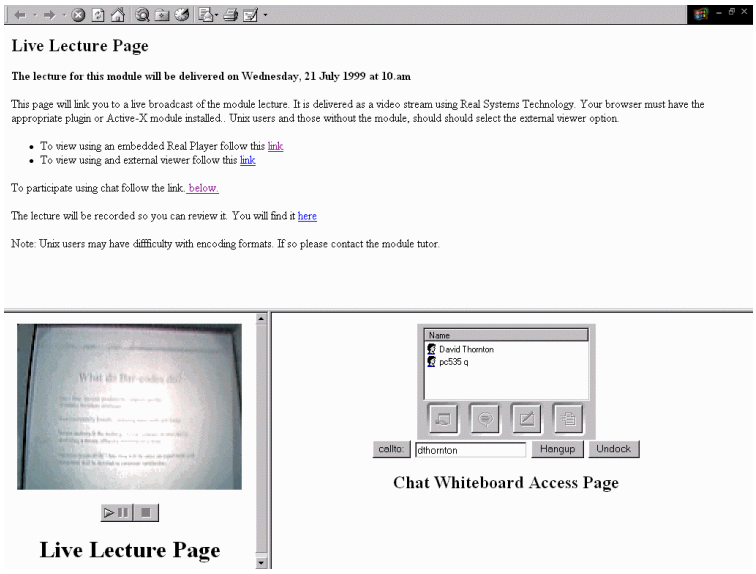

Figure 7.5 Online Live Lecture View Showing Frame Presentation of Embedded Video and Conferencing Controls

Figure 7.5 shows how a new frameset template was used to control the presentation of textual information, and to embed the video player and specific NetMeeting controls in *html* pages. Once again, ActiveX and embedding tools were used. Students could join a lecture from the browser. Figure 7.6 shows the result of having done so. In this image, not only are the previous controls still visible, but so too are:

- A chat window providing a synchronous share text work area for all online students and the teacher.

- The tutor's display. The image quality afforded by the camera was inadequate for showing projected images (figure 7.6). NetMeeting's application sharing was used to share both an electronic whiteboard and a PowerPoint presentation. It can be seen, that these applications appear in their own window and that tutor control over the presentation format has now been lost. Window sizes greatly influenced the area the shared screen displayed and large format student monitors may be required to gain full benefit from the technology. Application (data) sharing is a powerful tool, but requires careful use on the parts of both teacher and student.

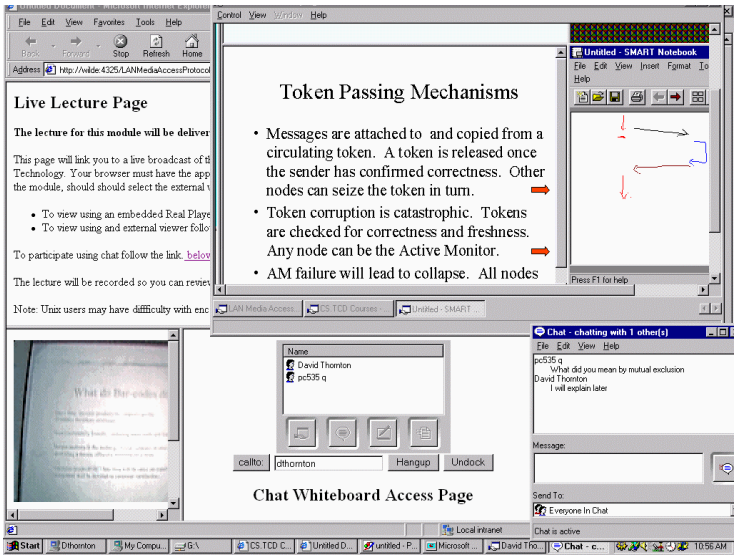

Figure 7.6 Student View of Online Lecture Showing Multiple Window Presentation and Video Quality Issues

Streamed video was archived at the server for review purposes, however the latency of up to 15 seconds was significant when used in conjunction with other truly synchronous applications. CuSeeMe can host multicast conferences and was used to provide synchronous video group conferences in the group discussion lesson. The conference was joined from the web page using a metafile, as described in the last chapter, however the video could neither be archived nor the interface embedded. Up to 12 video channels could be displayed simultaneously in group conference mode. If both synchronous delivery and archiving is required then infrastructural support for both, and parallel audio and video capture may be necessary. CuSeeMe did not provide application sharing directly but used the NetMeeting tools described above.

7.7.3 Capturing and Digitising Video from Tape

Video is an effective media for supporting learning. A video clip was captured from tape, digitised and converted to RM5.0 format. The film source "Lord of the Flies" was in PAL format on VHS video tape. Audio-Visual and Media Services, Trinity College, captured the clip and provided three QuickTime (*mov*) clips with different frame formats.

The 89-second video clip was captured using a video recorder. It was then passed to a Video2000 in-line editing application on a Mac platform for encoding to *mov*. QuickTime is one of the Real Encoder input file formats. The video data was not compressed. A capture rate of 25 frames per second and frame sizes of 320x240, 192x144 and 176x132 were specified. The *mov* files were further processed using Media Cleaner Pro with resultant file sizes of 498, 182 and 153 Mbytes respectively. Unfortunately, the *mov* output from Media Cleaner was not compatible, so Quick Time Pro 4 was used to convert the smallest file to *avi*. (compressed to 19Mbytes). This file was then encoded for streaming at 100kbps and 28kbps. The encoded files were saved on the server. Their final sizes were 1.1Mbytes and 236 Kbytes respectively.

The Production of video clips appropriate for different network conditions was possible only because the encoding process started with an uncompressed source. If video is to be used effectively, specialist human and technical resources are required.

7.8 Summary

In this chapter we saw how the open unified but distributed architecture identified in the last chapter provided an effective framework for building courses. Structural and presentation information stored in the database was easily accessible. The DBMS provided great freedom in how that information was used. Sharing permitted a collaborative approach to curriculum development. HTML pages were structured and generated on the fly and new facilities, such as administrative support and XML metadata querying were added to the system quickly and simply. Thus the architecture was both flexible and extensible.

HTML frames provided an open delivery vehicle for a wide variety of media formats and a means for controlling presentation at the client. Good tools were available to assist the author and the large variety of them gives teaching staff the freedom to use those with which they are familiar or comfortable. Servlets, SQL and *html* forms provided an effective mechanism for passing state information and for implementing adaptivity.

Collaborative tools were incorporated into the web page environment. They were both embedded and run from external windows. Their dynamic nature made presentation control at the client end difficult. The effective application of such tools required careful consideration, technical infrastructure. Students and teachers had to show flexibility in how their interfaces were used.

The prototype implementation set out to prove the concept that an unified distributed architecture can be used to design, develop and deliver academic courses making use of a wide variety of multi-sensory technologies. Furthermore, that the current state of of-the-shelf tools was such as to enable teachers to produce effective learning content and activities simply. This chapters has shown that the concept is sound and that the distributed approach also adds flexibility and extensibility to the system.

The final chapter will evaluate the prototype course produced from a technical perspective.

Chapter 8 Evaluation, Conclusion and Recommendations

8.1 Introduction

The objective for this dissertation was to establish whether current distributed technologies could be used within a unified architecture simply yet imaginatively in support of education. Their application would start from the point of developing a structured curriculum and extend to the point of finally delivering multi-sensory learning activities capable of meeting specific learning needs.

In the course of the document it was explained how a structured curriculum should be based upon sound pedagogical theory. It should comprise individual and group learning activities. These should have specific objectives and the teacher should offer a tailored environment to help the student achieve them. Creating such environments requires a palette of multi-sensory aids. Technological tools are available to help the teacher, but they will only be adopted if they are appealing. Any supporting architecture should allow the teacher to use his preferred tools. We saw how *html* is supported by many established tools and how *html* content can be linked simply using a URL to form an online course.

Most of the commercial courseware applications support *html* and structure courses using internal database files. They may also include templates to help with course development and specific utilities to support administrative and learning support functions. Templates assist with designing interfaces with which the student can feel at ease. The databases at the heart of most applications were tied to the application. Users could not extend the functionality of the application. Nor could many of the applications integrate seamlessly within an existing IS framework. Courseware applications include much functionality, however the creative freedom offered to the teacher remains bounded by the capability of the application itself.

In the last chapter off-the-shelf tools were used within an open distributed architecture to create a prototype academic course module. An ODBS compliant relational database was at the heart of the system. The curriculum was developed using its interface. Pedagogical requirements were set for each learning component in accordance with the course structure. Content was created using a host of different multi-sensory technologies,

each addressing a particular lesson objective and individual learning style, and online interactive and collaborative activities were developed to meet group learning requirements. Student registration was carried out remotely and class lists were compiled using a database form. The hypermedia course was produced on-the fly. Its structure was adapted to student requirements. Learning components were offered to the student via a personalised GUI. The presentation format, based on frames, provided a controlled environment for the concurrent display of several different multimedia learning components on a single screen. Collaborative activities were provided through the use of video and data conferencing.

In this chapter we will evaluate the architecture, the effectiveness of tools support and the benefits provided by the unified approach; the preceding material will be reviewed with hindsight and condensed as a conclusion; recommendations for future work will be presented and finally a few words as to the significance of the study to IT support for education.

8.2 Courseware Evaluation

Several criteria were presented in chapter 5 for evaluating courseware applications. These will be used as the basis for evaluating the implementation.

8.2.1 Simplicity of Content Creation

Static content was generally easy to create. PowerPoint slides were converted to HTML without assistance. Furthermore, its vector graphics work area was used to produce effective *gif* and *jpg* images for web delivery. The interface for the Dreamweaver *html* editor was intuitive. Draggable controls and menu commands were used to generate text, tables, interactive *html* animations and embed audio, video and other third party content. Its template library supported delivery over Netscape and IE clients .

The production of quality universally playable audio and video content was less straight forward. There was a limited availability of low-lost cross-platform video capture applications for producing uncompressed video. Windows NT4.0 was particularly poorly supported in terms of video cameras. Producing streamed video from live or archived digital video using NetShow, RealProducer or RealEncoder was not difficult. These application also generated the accompanying metafiles. The latency associated with streamed video made it inappropriate for being combined with other synchronous

collaborative activities. Used in isolation, however, it was an effective mass delivery medium for lectures, for example.

Capturing video from tape and encoding it was time consuming and required specialist resources. However it facilitated the production of multiple-format video clips. The process could be streamlined to meet increased demand.

8.2.2 Tools Palette for Managing Interactive Multi-sensory Content

The tool palette was *html*. It proved to be a very effective vehicle for carrying other multi-sensory content. Despite scripting problems, most multimedia content included in the prototype was be rendered simply. This provided the course developer with an enormous array of possible learning material. Tools support for the palette was good and afforded much flexibility in how lessons could be shaped to satisfy the teacher's objectives. Abstract course components from the curriculum were realised through the database. Students were provided with URL maps to assist them with navigating the content. These directed the student to the URLs of suitable content or to front web pages hosting live lectures or discussion groups. the teacher. The utility for XML (IMS) metadata querying, which was added to the application, provided a mechanism for locating and reusing suitable content from elsewhere. This utility was not found in existing commercial applications. The Access form was adequate for the task of resource mapping, however the quality of the interface could have been improved considerably. A web interface could also have been provided, however the experience gained from using commercial courseware applications indicated that such an approach may be much slower and more cumbersome. A distributed drag and drop interface would be consistent with other productivity tools.

8.2.3 Interface Quality

The prototype reinforced the importance of good interface design for developing online courses. This dissertation set out primarily to address issues relating to the exploitation of a distributed architecture and the application off-the shelf tool rather than interface design specifically. The quality of the interfaces provided for students and teachers reflects this. The policy of support for off-the-shelf authoring tools encouraged the teacher to use those with which he was comfortable. The only tools which the teacher was required to use were the course structuring and resource mapping forms. Frames were an effective

presentation vehicle. Only three formats were provide and these were hard coded, A larger variety of frameset templates could have been made available to cater for different environments. The ability to resize frames was very useful.

8.2.4 Facility for Synchronous Communication and Collaborative Workspaces

Conferencing was supported effectively by the architecture. However, the dynamic nature of the activity and variations in display settings made it impossible for the teacher to control presentation on client interfaces. Embedding the NetMeeting ActiveX controls in a Web page improved the appearance of the application GUI. However, the clients it spawned were all presented in their own windows. All CuSeeMe clients were launched as windows applications.

CuSeeMe in multicast conference mode was suitable for delivering lectures to large groups in simplex mode. The latency issues associated with live streaming did not apply. CuSeeMe could be combined effectively with NetMeeting for data (synchronous duplex) and video conferencing. The MeetingPoint conference server was not evaluated. Group conferencing (duplex) with up to 12 clients was adequate for student discussion groups, especially combined with data conferencing for application and graphics sharing.

Although the technology was adequate for the prototype, its implementation was primitive. The hardware available for it would have been inadequate for any rollout implementation. Much greater planning and preparation would have been required to support a real virtual class presentation. Class rooms would have needed to be wired for audio, video and other technologies such as electronic whiteboards and document projectors. Training for students and teachers in how best to managing virtual environments would also be essential. The virtual class room represents a transformation and best practise should reflect it.

Combining synchronous with streamed media was not successful due to latency issues. Mechanisms for synchronising events within *html* frames would have been useful. JavaScript, in particular it's facility for event handling within windows and documents, was not exploited fully. Together with metafile support, it could have addressed the problem. Further investigation into the possibilities afforded by the NetMeeting SDK may suggest other ways for creating truly integrated interfaces, albeit on a Windows platform. The conferencing tools were exclusively Windows-based. This is a weakness

in the prototype implementation. However, it is believed that the JMF will lay the ground for a cross platform solution.

8.2.5 Support for Cognitive Structures

Cognitive support facilities were limited. The class lecture was recorded for review. The course was structured to support learning style and hierarchical lesson objectives. However, no support was provided for note taking, book marking, glossaries or context sensitive help resources. This was considered beyond the scope of the study.

8.2.6 Support for Student Experimentation, Simulation, Auto-correctable Tests.

No specific facilities for experimentation, simulation or auto correctable tests were provided. Other work has demonstrated how JavaScript can be used to create interactive auto-correct multiple choice tests. Elementary student tracking was provided through passing user-names. This could have been used in support of testing. However, it was not exploited by the application. The proven flexibility afforded by the database has convince the author that including and integrating test management would have been possible. Sophisticated tracking would have required that content as well as structure be controlled by the hypermedia engine. The potential for using JavaScript to facilitate this was not exploited.

Simulation was provided through the use of interactive animation. Macromedia Flash was a very effective tool and well suited to producing cross-platform animation and graphics for the Internet.

8.2.7 Strength of Course Management Capabilities and Breadth or Reporting Tools

Course management was provided at a basic level. Students were registered with the institution, included in class groups and given access to learning material and class activities on that basis. Students could be members of more that one class. Testing was considered outside the scope of this prototype implementation. Thus the student reporting utilities were not included.

8.2.8 Degree of Integration with Curriculum Development Systems, Support and Administrative Services

Course delivery was intimately integrated with the curriculum development system. The prototype used a three level structure (Course, Module, Lesson) for defining course components. It could easily have been extended to bring in College, Faculty, Department….. The essential function of the architecture was to bind the curriculum to the learning material developed to meet its requirements and to the administrative services supplying the students and academic staff.

The shared access to the database provided for collaboration amongst academic and administrative staff. The visibility it afforded could have had the potential to transform the faculty. For example, changes made to course objectives at module level could be immediately visible to administrators developing promotional literature. Microsoft Access would not be the RDBMS of choice for enterprise rollout. Oracle, an established enterprise application, would scale better and could be used in lieu without significant modification. Its JDeveloper product includes many web integration tools

8.2.9 Flexibility of Deployment Methods

The architecture identified in this paper was selected on the basis of it being network based and supported by TCP/IP technology. Within that constraint, support was provided for delivery over high speed LANs and remote dialup networks. This took the form of the development of a suit of bandwidth-sensitive content supported by adaptive presentation and metafile access to the content. Support for CD-ROM deployment was not included. Conferencing was evaluated within an Intranet environment. Proxy-server and firewall issues would have to be addressed for exploitation beyond the campus.

8.2.10 Cost of Ownership.

The adoption of a distributed architecture permitted its exploitation at very low cost. The Java API's, JRun, Apache, RealEncoder, Real Server, Microsoft's Data Access SDK, NetMeeting, Internet Explorer and Netscape Communicator were all downloaded from the Web at no charge. Using a DBMS, for which a license is already held, also incurs no additional expense. For many institutions this may be Oracle, for other small organisations, this may be Access installed as part of Microsoft Office. Thus the

distributed approach is not only more flexible that that afforded by specialist courseware application, it is also cheaper.

The main expense lies in the provision of content creation tools. However this is also the case for most other *html*-based courseware applications. Freeing the institution of the cost of supporting the architecture permits the resources to be directed towards those content authoring tools which appeal to the teachers themselves, be it PowerPoint, Dreamweaver, Flash, Corel Draw (vector graphics), Adobe Premier (video in-line editor) or any other of the plethora of *html* compliant productivity tools on the market. Java's continuing development will, doubtless, also make available powerful low-cost multimedia and conferencing application in the near future.

8.3 Conclusion

There is widespread use of courseware to deliver education and training. Many applications provide tools to assist teachers with the development of learning structures, with integrating learning material, with managing student groups and other administrative and learning support utilities. Most applications are built upon a closed architecture with few opportunities for integrating course development with other components of the institution's education support systems. The ready availability of low cost multimedia, conferencing and other distributed technologies questions the need for monolithic solutions. Courseware applications have taken this on board and provide tools for migrating their content to Web formats. This dissertation has addressed the question of whether the functionality provided by the applications themselves may not be equally well served by a distributed solution.

Pedagogical theory laid down the requirement for structured learning. This may be presented in the form of a formal curriculum. It was shown that a relational database can be used to develop a curriculum structure. Furthermore, a modern RDMS provides the tools for collaboration in the curriculum development process and for controlled visibility to its information across the Web. Relational databases are an established part of an organisation's IT structure and functional co-operation is commonplace. A courseware application centred on an ODBC compliant relational database is more extendable, scaleable and flexible than one using proprietary data structures.

HTML content can be presented using Web browsers and can be accessed using an URL. There is wide tools support for developing multimedia *html* content. The openness it

affords provides a mechanism for serving multi-sensory learning material and activities from a web page. The material can be mapped to a database curriculum entry so affecting an integration between course delivery and the curriculum development systems. Middleware applications can be developed easily to extend the functionality of the hypermedia course delivery engine. Course structures can be generated on the fly in accordance with a hierarchical curriculum or by the imaginative use of metadata such as afforded by XML(IMS). Student tracking can be used to monitor his progress through the course domain and provide cognitive support and reporting tools.

HTML and the role played by a resource's URL is central. Teachers require lesson authoring tools which enable them to present learning material in a format which suits their message. They need tools to work with text, images and audio-visual content and to assist with collaborative tools. It was shown that a large variety of tools exist, many well established, to do this and that their results can be accessed using a URL. Thus *html* provides a creative palette without imposing constraints on the teacher.

A unified distributed architecture was identified and used to build upon the integration of curriculum development, course authoring and the creation of multimedia content. Functional partitioning was introduced. Web servers were used to serve static content, streamed media servers provided bandwidth-sensitive assess to live and archived compressed video, conference servers managed group and multicast conference activities. Each server was able to address the particular needs of its particular media type and access to it was be provided from a web page using its URL.

Java and servlets were shown to be easy to use in developing the middle support for the hypermedia engine. JavaScript was shown to be an important component in authoring integrated interactive lesson pages. Tools support for it exists. However, the capability of JavaScript extends beyond that exploited in the prototype course implementation. Neither, was applet technology exploited. Client side applications, such as plug-ins and ActiveX controls, were used to present many multimedia components. They were platform sensitive; cross-platform Java-based players are not yet sufficiently profligate to support true openness across the architecture.

Specialist resources were required to assist with producing network and platform sensitive video multimedia content. The design of user interfaces is non-trivial and benefits from specialist assistance.

8.4 Recommendations

- That a research be conducted into the practicalities of authoring multimedia components on demand together and managing and accessing a learning component repository.
- That to potential for developing platform independent multimedia and conferencing tools using JMF, for example, be established.
- That the educational benefits of using a collaborative database tool for curriculum development be assessed.
- Further research should be conducted into how students and teachers can adapt to the virtual classroom. Are current HCI standards for interface design appropriate to dynamic page content? How can scripting help the end-user rescale window and frame presentation to suit his own platform requirements.

8.5 Final Remarks

The main achievement of this study was to show how low cost off-the-shelf application software could collaborate within a distributed environment to provide comparable functionality as is afforded by state of the art courseware applications. Moreover, that the flexibility which comes from its distributed architecture and the openness associated with using *html* and an ODBC compliant database provides the potential for functional evolution and integration, consistent presentation and authoring freedom. Thus, any institution which uses such a database for curriculum development and course administration has laid the foundation for a distributed adaptive hypermedia courseware application.

Annex - Script Segments for Embedding Plug-ins and ActiveX Controls

This annex includes HTML scripting to assist with embedding multimedia content. It should be used as guideline only.

Segment 1, below, was used to embed a *avi* movie in a web page. It demonstrates the use of support for ActiveX controls and a corresponding plug-in player. The segment begins with the option to download the movie and play it using the default external viewer. Then the <object> tag defines the embedded control.

First, details are provided for the ActiveX control. In this case it is the Windows media player. The classid conforms to the Windows registry entry, so if it is already installed, the movie will be played immediately. If it is not installed a codebase can be provided so the control can be downloaded and installed seamlessly. The use of a codebase tag simplifies the installation process. An id is provided for the movie object and parameters are given for setting the configuration for the media player. The movie is embedded inside the object tags. In this case a plugin application is available and download instructions are provided. Plugins are configured to run with the browser, and generally require user action to install them. The process is not seamless.

In the segment 2, a NetMeeting ActiveX control is embedded in data mode only, that is without the video viewer. Note, how buttons can be included to handle events. In this example, the address of a conference server is passed to the NetMeeting application when the user clicks on button2. In this example, unfortunately, the codebase is not included.

The final segment was used to embed the *swf*v4.0 animations in to the hypertext document. This time, the code base is included. For the codebase or plugins page to be effective, users should have administrator rights on the host, which may not be desirable on enterprise Intranets.

Segment 1

```
<HTML><HEAD><TITLE>Windows AVI Presentation</TITLE></HEAD>
<BODY ><H1>The Conch from The Lord of the Rings</H1>
<p>If you prefer to use an external player, please click <a href="conche_176_132.avi">here</a>
<object id="MediaPlayer"
        width=384 height=282
        classid="CLSID:22D6F312-B0F6-11D0-94AB-0080C74C7E95"
        codebase="http://activex.microsoft.com/activex/controls/mplayer/en/nsmp2inf.cab#Version=5,1,52,701"
        standby="Loading Microsoft® Windows® Media Player components..."
        type="application/x-oleobject">
        <param name="FileName" value="conche_176_132.avi">
        <param name="ShowControls" value="True">
        <param name="AutoRewind" value="True">
         <param name="AutoStart" value="True">
        <embed type="application/x-mplayer2"
                pluginspage="http://www.microsoft.com/Windows/Downloads/Contents/Products/MediaPlayer/"
                src="http://wilde.cs.tcd.ie:4325/LANMediaAccessProtocols/conche_176_132.avi" filename="conche_176_132.avi"
                width=384    height=282
                showcontrols="True"
                autorewind="True"
                autostart="True"></embed>
</object>
```

Segment 2

```
<object id=NetMeeting
        classid="CLSID:3E9BAF2D-7A79-11d2-9334-0000F875AE17" width="373"
        height="168">
        <param name = "MODE" value = "DataOnly">
        <embed mode="DataOnly"
                src="LANMediaAccessProtocols/DataOnly"
                width="373" height="168"></embed>
</object> <br>
<input type=button  value ="callto:"id=CallToBtn
onClick=NetMeeting.CallTo(CallToAddress.value) name="button">
<input type=text  id=CallToAddress  name="text" value="pc535.cs.tcd.ie>
<input type=button value ="Hangup"id=HangUpBtn onClick=NetMeeting.LeaveConference()
name="button2">
```

Segment 3

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://active.macromedia.com/flash2/cabs/swflash.cab#version=2,0,0,0"
width="386" height="280"
align="right">
<param name="SRC" value="movie002a.swf">
<param name="PLAY" value="true">
<param name="LOOP" value="false">
<embed src="movie002.swf"
        pluginspage=http://www.macromedia.com/shockwave/download/
        width="386" height="280"
        align="right"
        play="false" loop="false"></embed>
</object>
```