

# Easy-to-use multimedia tools and scalable distributed architectures for web-based teaching and learning

Professor Chris R. JESSHOPE  
Director NZEdSoft, Massey University, Private Bag 11222, Palmerston North, New Zealand  
and  
Department of Computer Science, The University of Hull  
Hull, HU6 7RX, UK

## ABSTRACT

paper describes work being undertaken at both Hull University and Massey University in the development and deployment of tools for web-based teaching and learning. The tools described here comprise an easy-to-use multimedia authoring system and a scalable system for the management, authoring and delivery of education in both on-line and off-line modes. The former is the AudioGraph Recorder and is in use by a large number of teachers and trainers around the world. The paper will give a brief introduction to its capabilities, describe the work in progress and show how it has been used in practice. The educational delivery system is designed to be both flexible and scalable. To this end we have exploited both legacy and emerging standards in database technology, such as SQL and XML. The system is also designed to be scalable by utilising the network of students' computers as a distributed server. The architecture and implementation issues solved in developing this system will be discussed.

**Keywords:** Multimedia authoring, web-based teaching and learning, Web-based education management and delivery.

## 1. WEB-BASED EDUCATION

Web-based education has been held up as a panacea for many years; our own work in this field started in 1995 and from very practical requirements. The initial reluctance to adopt is changing and we are now seeing a rapid take-up of this technology. A good example of one of the drivers is the UK initiative of the e-University[1]. This is typical of the widespread interest of governments and institutions in this use of the Web, together with the ubiquitous personal computers, to enable information to be exchanged more quickly and flexibly than ever before. People now expect information that is immediate, personal, current, engaging and collaborative[2].

Technology and knowledge are also the momentum of any new economy. There is always a shortage of skilled workers in the work force. Post-secondary or post tertiary education is now mandatory in the workforce due to the rapid and accelerating rate of change of many sciences and technologies. This requires a constant refreshing of knowledge in a skilled work force. This education will have to be provided by many new players and, in addition to the traditional schools and universities, new education providers, including the companies requiring the skilled workforce, will be brought into this new arena. This education may be provided locally, on a company's Intranet or may provided globally, with experts in a given niche selling their expertise to the world. The technology for this already exists, using any number of different delivery methods[3].

The list below includes only those relevant to web-based delivery, the most promising mode of delivery due to its widespread use.

- Compressed Video
- Audio/visual presentations
- Audio
- Electronic document exchange
- E-mail, Usenet
- Electronic classroom environments
- Internet or CD-ROM/DVD delivery

Web-based delivery of distance learning material can be provided synchronously (in real-time) or asynchronously (by recording and serving the material on a web site). Media such as audio, video, image, text and graphics can also be delivered in an interactive and above all a continuously monitored environment and the cost is affordable for all learners, regardless of their location.

Today's web browsers such as Netscape and Internet Explorer are so easy to use that even a computer illiterate person can grasp the basic skills in a very short time. These browsers are free to use and their functionality can be extended by either Java applets or by providing plug-in components.

This delivery mechanism is also dynamic, as it provides the learner with an interactive interface, integrating all kinds of media within a hypertext environment. "*The WWW is immediate, personal, current, engaging and collaborative. Learners are responsible for their own learning in a medium that allows exploration and discovery*"[2]. Web-base learning therefore helps the learner to develop skills such as how to collaborate and how to find knowledge from the extensive resources available. Validation of that knowledge is one of the major hurdles for the adoption of this direction in education. We will return to this issue later.

There are now significant political pressures to ensure the needs described above are met and education providers are pursuing technology to achieve these goals at an ever-increasing pace. It is predicted[4] that in the United States over the next two years, colleges and universities will use web-based learning as a means to improve education flexibility and to provide education opportunities to more students. With only 30 percent of classes now using web pages to distribute class materials and resources and less than 5 percent of campuses having adopted web-education delivery platforms, the adoption is still in its infancy. However, it is estimated that within the next two years, about 50 percents of all campuses will install a campus-wide learning platform.

## 2 . DEVELOPMENT AND DELIVERY OF EDUCATIONAL MATERIAL

Searching the web reveals many thousands of web-sites that purport to deliver educational material. However, these so-called "web-based distance leaning systems" do little more than put lecture notes online or just create links to material. Although this is may be useful to augment a traditional education, it is not sufficient in itself to provide an on-line education and is certainly not exploiting the medium to its fullest. There is in fact a dearth of high-quality educational material. Even sites that contain animation, graphics, video or audio, tend only to use this approach for visual effect, without regard to pedagogical efficacy.

Concerning the delivery of this material, there are currently two approaches: to use a standard web server, with or without access restrictions, or to use one of the many educational web frameworks. These will include access checking, access logging, on-line communications, and perhaps some simple authoring e.g. for tests. Although most of these systems are very similar, there are numerous comparative evaluations to be found[5-8].

Given the extent of the material and tools available, why is there still a need for continuing development in this area? One of the reasons that most of the material available is text-based is due to the high cost of developing multimedia. This may require hundreds of hours preparation for every hour of presentation. This ratio is untenable and it is clear that the development of the media must be simplified in order to reduce the costs and at the same time bring the educators into the picture. AudioGraph[9-11], one of the tools described in this paper, has attempted to do just that. This tool will be described briefly in the next section. It has also been demonstrated that it can be used successfully to produce and deliver economical educational multimedia[12,13].

Returning to the web-based delivery of educational material, the existing tools[5-8] have been developed as ad-hoc manner extensions to a conventional web server. Most are based on server-side scripting, with Java providing for client side interactivity. None of these tools really provide standards for interfacing to legacy systems or for interoperability across other learning objects that can be found on the web. In our analysis we found the following deficiencies:

*Lack of flexibility in the mode of delivery:* all provide only online delivery. This can be both expensive and inflexible. All material has to be downloaded from the server rather than delivered on CD or DVD. In the case of video or even high-quality audio, this requires end-user bandwidths unavailable in most of the world. The problems that need to be solved in allowing an off-line access mode however, are not inconsiderable.

*Lack of flexibility in interfacing with legacy systems:* many current education providers already have large investment in database systems for administrating their business. Web-based systems must integrate seamlessly into these legacy systems, which will usually require SQL database interfaces.

*Lack of adaptability:* current tools are mostly designed for browsing and information retrieval, but not for active learning. Students with different background knowledge will essentially

receive the same educational materials. Adaptivity in delivery is an important aspect of web-based learning systems. It is defined as the ability to be aware of user's behavior and knowledge, and to take this into account in providing the user with the right kind of learning object[14]. There is a conflict between this and off-line delivery requirement. On-line access can be monitored easily but in off-line mode, with material delivered on CD, special consideration has to be given to the logging of student activity and its use in adapting the presentations the student receives.

*Lack of accuracy in multimedia document retrieval:* querying delivery systems is normally achieved based on keyword searches. Accuracy of keyword searches is not good, normally this would yield either to few or too many matches. Semantic-based searches have the potential to provide much more accurate retrieval.

From this analysis, it is clear that there is also a need for further development of web-base distance learning systems to overcome these shortcomings. This is the origin of TILE project [15].

Both the AudioGraph and the TILE system will be described in this paper. Work on the AudioGraph is very mature with many users of this commercial-quality software. This project is described in section 3. The TILE project, on the other hand, is currently in the development stage. The design stage has been completed and the result of that is described here. We have already prototyped most of the system and will describe this in section 4.

## 3. THE AUDIOGRAPH TOOLS

AudioGraph was developed for on-line teaching and training. The tools comprise an authoring program called the AudioGraph Recorder and plug-ins for web browsers to play back the recorded presentations. The tools are available as a free download from <http://www.nzedsoft.com/>. Currently the Macintosh version is more mature, with version 1.3, the fourth release, having been made available in mid 2001. The Windows version, released at the same time, is a beta release (version 1.0beta). This section describes the tools and their planned further enhancement.

The fundamental philosophy in the design of the AudioGraph has been to assume that the tools would be used by academics, teachers and trainers, rather than by multimedia professionals, who are already catered for by a number sophisticated and complex authoring tools. This required a thorough analysis of the basic requirements of on-line education and the provision of an intuitive interface that allows these non-professionals to produce professional looking presentations very rapidly and without prior multimedia experience. This strategy places the development of multimedia firmly in the hands of the educators rather than the professionals. In this way, we hope to achieve a paradigm shift, in much the same way that paper-based publishing underwent with the advent of easy-to-use word processors.

At the outset of this project our goals were as follows:

- to target the delivery of multimedia course-ware via the world-wide web;

- to provide simple-to-use tools capable of producing multimedia presentations with little or no experience of multimedia editing;
- to significantly reduce the industry norm of 200 hours of preparation time for every hour of multimedia presentation;
- to ensure the presentation size was small enough so that institutions could provide a large corpus of multimedia tuition on a modestly sized server;
- to ensure that the multimedia presentations could be accessed over modem connection speeds (e.g. 14K), without any significant delay;
- to provide cross platform delivery of the multi-media contents using a browser plug-in.

The fact that these tools are now used by around 1000 users worldwide, is an indication that these objectives have been largely met. The tools are under constant review but we are resisting the temptation to add too many features to maintain ease of use.

### An overview of the tools

We had to make compromises in the design of the tools. The major one is in the media supported. AudioGraph supports only compressed audio, images and vector graphics as its media elements, and generates presentations for web-based delivery. We use an iconic interface to avoid void synchronisation issues using the AudioGraph principle, which requires the presentation to be a strict sequence of media elements. Some of the media elements have real-time semantics, such as pauses and sounds. Others, such as the rendering of an image or vector graphic component, have no real time meaning but are rendered at the speed of the playback computer, which may be very different on different generations of computers. Whatever the playback speed however, the media file structure, which embeds the AudioGraph principle, maintains the strict sequence and hence the correct semantics of the presentation.

AudioGraph does not support video and clearly this is to satisfy the requirements of a small footprint web-site and low-

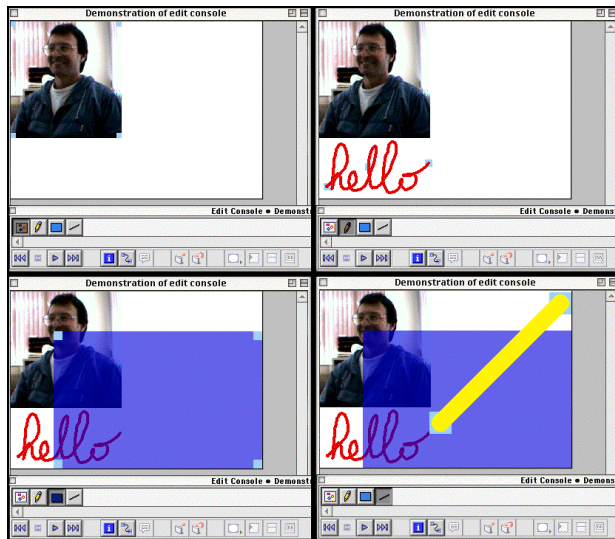


Figure 1. A composition of screen shots showing the edit console and slide window. This shows the correspondence between the iconic and graphical representation of each media element

bandwidth streaming media.

Results from using the tools in extensive field trials are presented in [12,13], which demonstrate the effectiveness of the tools.

### The editing interface

The main editing interface that implements the AudioGraph principle is a pair of windows: *the slide window*, which shows the presentation as it will appear at any point in the presentation sequence and the *edit console*, which represents the sequence of media elements in iconic form. The user can select any icon within the edit console and the slide window displays the presentation, as it would appear when that media element has been displayed. Figure 1 gives a demonstration of this principle. The four media elements in the edit console are an image, a freehand vector graphic element (the handwriting), a rectangle (notice the transparency) and a straight line, in that order. Notice that only those elements up to and including the icon selected in the edit console are displayed in the slide window. Any selected element can be moved in either window (in space or in time). The edit console shows the presentation as a sequence of media elements (iconically), which can be moved around on the temporal axis and the slide window shows a corresponding snapshot of the spatial representation of the presentation at any point on the temporal axis.

The edit console has a number of controls, which are for previewing the dynamic presentation. These navigate, play and stop playback. Other controls are used for editing and grouping together the media elements.



Figure 2. The tools menu

### The tools

Figure 2 shows the tool menu, which is used to select a tool to place the media elements in the presentation. At the top of the menu, there are various vector graphic tools, namely (from left-to-right): freehand line - used for handwriting, straight lines, rectangle (open and filled), ellipses and finally, arcs. Then there is a highlighter, used like a highlighter pen and an eraser. Below these are the image placement tool and the link tool, then the scroll and the text tools and finally the sound and pause tools. The remainder of the window provides some tools for editing attributes and navigation. If a media element has been selected, then these tools modify its attributes, otherwise the attributes of the tools themselves is edited. In this window, colour and line thickness can be modified, which are the most

commonly used attributes. Other attributes can be edited from a separate attributes menu.

**The media elements**

**Images** - A simple presentation style using the AudioGraph would typically use a set of presentation images, as might be produced by PowerPoint for example, which are then annotated using audio, highlighting, handwriting graphics and possibly other images. This is not the only way that a presentation can be constructed and [12] gives examples of a number of different development techniques.

Images can also be imported from a screen capture, or cut or copied from another application. AudioGraph supports transparency in its vector graphics and when this is combined with the pixel-by-pixel transparency of the PNG graphics[17]

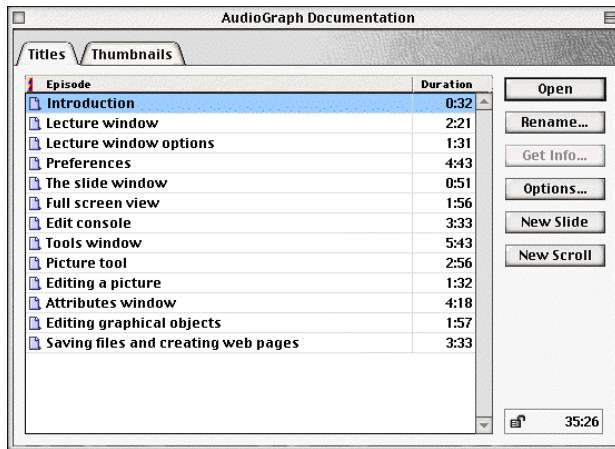


Figure 3. The Lecture window, showing the characteristic of each slide in a presentation, including duration of slides and total presentation.

used in the web presentations, some very sophisticated results can be achieved in a presentation.

**Sounds and pauses** - The most data-intensive media element is sound. Fortunately there are very efficient compression algorithms for transmitting sound over a network. This is especially true if, as is expected, the sound input is going to speech. The AudioGraph uses the GSM compression technique [16], which is optimised for speech. An uncompressed speech stream of 16-bits accuracy and sampled at 8KHz would require 128 Kbits/sec for transmission over a network. When compressed using GSM, the 16-bit/sample speech would require only 13.2 Kbits/sec thus providing a compression ration of approximately 10 to 1. Additional compression is achieved by dividing the stream into sound bytes separated by explicit pauses of arbitrary duration. Sound can be recorded directly, using the sound tool and a microphone. Sound of any sample rate and precision supported by QuickTime can be cut or copied from other applications and pasted into the AudioGraph using the clipboard. Compression takes place, as with images, on generating a web site.

**Websites**

AudioGraph builds quite sophisticated web sites, without any web editing at all. The basic structure of the site comprises an

index page of links, one for each of the presentation units (we call them slides or episodes) within the lecture document. Figure 3 shows the lecture interface, which lists all slides within a presentation. It allows the user to specify a meaningful name for each slide, lists the duration of each slide and the total duration of the lecture.

This window reflects the structure of the web site that will be generated for this presentation. Each slide will generate a link to it as a presentation. That presentation may be generated within the same window or may use a new window. The Options button allows various attributes of the slide to be set, such as size, background colour etc.



Figure 4. Web page index generated from the presentation given in figure 3.

A presentation was generated from the lecture shown in figure 3. The index page produced is shown in figure 4 and it can be seen that it mirrors the lecture window structure and also includes instructions for playback, including the download of the plug-in.

**Further developments**

We are still developing the AudioGraph tools. Our current version produces media, which is sequential, like a movie. The only control available to the student is the ability to select the component of the presentation from within the html index page and to position the playback position from within the plug-in interface. Although this provides all of the navigation that is necessary, the effect is not as seamless as if the links were made from within the presentation itself rather than the embedding html page.

The next feature to be added will be the ability to activate objects within the media presentation as links, either to other AudioGraph presentations or to arbitrary URLs. This will allow users to generate adaptive presentations, where the presentation seen by an individual student will depend of the selections he or she has made. In effect the student will flatten a graph of nodes, each of which is an AudioGraph episode into their

personalised presentation, based on the choices or answers they have given. In this way we will provide streaming presentations within which the students make committed choice.

This has been designed without compromising the already simple interface. Each episode will remain a strictly sequential presentation and the difference in interface will be small. A link attribute will be introduced at the slide window level and a new pane at the lecture window will summarise the link structure and allow the user to edit it.

Thus the change, which is quite a significant departure from the current tools, will be affected with minimal change to the interface and yet at the same time will solve a major problem in multimedia streaming that provides scalability of streaming in the presence of choice.

Other features to be added will include compressed high-quality sound, the ability to add typed text as well as hand-written text and the use of volume activated detection of voice recording in order to reduce the bandwidth of captured voice still further.

#### 4. THE TILE WEB-BASED EDUCATION DELIVERY SYSTEM

The TILE system has been designed to address the deficiencies in current web-based courseware management and delivery systems. If we consider the web browser as the first-generation of web-based educational delivery tool, then the tools analysed in section 2 might be considered as second-generation delivery tools. Many have evolved from universities' local solutions to the campus-wide delivery of on-line education. However, these tools have not really considered the implications of globalisation, which requires dependable and scalable distributed systems. Neither do these solutions offer a great deal of flexibility in terms of the delivery to the student.

TILE is a third-generation learning environment that must match the future we have outlined in section 1 and provide a robust and scalable solution to education delivery. Moreover, it must also meet the flexibility requirements of this new generation of learners, who will not be confined to our university campuses. TILE is a large collaborative project[15].

Let us consider the student's requirements first. The student may wish to study in a number of different situations:

- at home using their home computers - we assume that they have on-line access but that there is a finite cost for that access (it may be a time-based cost or, in the future, a packet-based cost where they are on-line continuously). Either way, we assume there is an advantage to reducing either cost;
- while travelling, with a laptop or similar portable computer - in this situation the student may have no access whatsoever to an on-line connection, or if they do it may be very expensive;
- at a conference, an Internet café or in a laboratory, using a computer, which they do not own and may not install software on..

This requires the support for both on-line and off-line modes of

delivery, which in itself is not a difficult problem, it only becomes a problem when we also consider the requirements of the educator:

- the lecturer will probably publish the course once per academic period but may also wish to publish an update of it while the students are studying. If the material had been delivered on CD ROM for off-line access this would cause of problem of version control;
- the lecturer may provide adaptive material, in which case some kind of access logging must be maintained locally and this again is incompatible with off-line CD ROM access.

We have been searching for a single integrated solution that captures all of these conflicting requirements.

Another issue is scalability, we must be able to distribute data intensive material to the student cheaply. If we use on-line data delivery, scalability means providing faster and faster servers and eventually creating networks of servers, with their associated cost. The computer scientist E.E. Dykstra once said of computer networks "never underestimate the bandwidth of a truck-load of tapes". Well, today it would be CD and tomorrow DVD but certainly we should not underestimate conventional distribution channels, such as posting material on CD or DVD to the student; this is eminently scalable. The characteristics of this approach are a high bandwidth with a relatively high delivery time (days instead of seconds) and of course, static data. Provided that we can overcome the inflexibility of having static data, this is a very acceptable solution. Static data is not such a large problem provided that we can allow producers to update the material distributed without publishing a completely new CD.

TILE allows the storage of material on CD or DVD but this material is linked dynamically using standard database techniques. Thus if a module were republished, version control

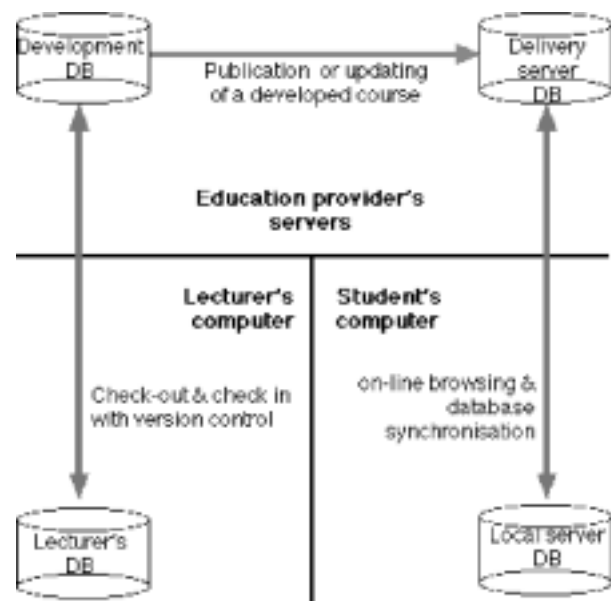


Figure 5. Courseware development and delivery cycle and the databases involved.

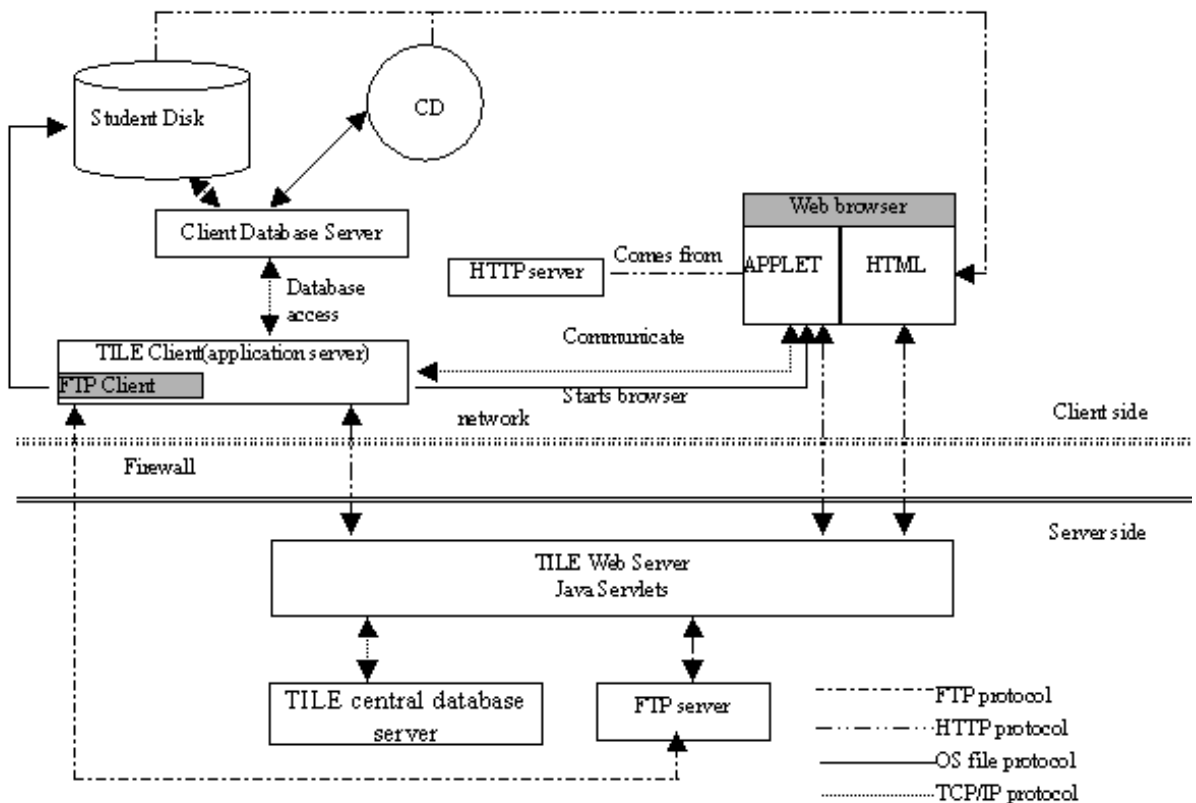


Figure 6. The TILE system architecture

in the database would redirect the link to wherever the module was located. This would initially be on the central server but may be downloaded onto the student's computer.

The second issue with off-line access is adaptive delivery. For pedagogical reasons, we may wish to monitor the student's progress in learning that material, building up models of their knowledge and preferred methods of learning [18,19] and then use these models to adapt the delivery of material on a student-by-student basis. This would not normally be possible unless the student was on-line. TILE allows adaptation in both on- and off-line modes by monitoring and logging students' choices and answers at all times. This also requires distributing the database that manages the course structure as well as the models held about a given student.

Figure 5 gives an illustration of the cycle of courseware development and the databases, which must be maintained and synchronised in order to implement the above solutions.

In this paper we will look in detail at the delivery system (RHS of Figure 5). This provides a number of different modes of access for the student to the educational material:

- *On-line with local server* - in this mode, most of the educational material is provided from the student's local computer having been distributed on CD or DVD. Additional software (the local server) is required on the student's computer to manage and monitor its delivery. The student still has access to the on-line archive on the

education provider's remote server(s) and local information can be updated from this source.

- *Off-line with local server* - in this mode the student has no access to on-line material, but the local server software continues to manage and monitor the student's progress.
- *On-line with no local server* - in this mode, the student for whatever reason is unable to install the local software and must access all material from the education provider's remote server(s).

In the latter two cases, inconsistency may arise between the information on the student's computer and that on the education provider's remote server(s). This may be due to the publication of new material on the remote server or updates in the student's logs or models. In either case, TILE implements mechanisms to synchronise inconsistent information between server and student databases.

#### The file architecture

**A client-server approach** - The architecture of the TILE delivery system is a client/server one. We use a thin client to present the information to the student and the server keeps track of and structures the information to be delivered to the student. The lecturer or content provider will define the structure, using an authoring interface tool (LHS of Figure 5). This may be modified by the students' models or indeed, may be overridden by the students themselves. In order to provide such flexibility, the system is database driven, using SQL for the students' and education providers' servers. Data exchange

between the databases and the database on the lecturers' computers use an XML representation of the structure. The flexibility provided by the database allows for the distribution, updating and synchronisation of material.

Different modes of access have forced us to modify the traditional client-server architecture. In effect we distribute the server function to the students' computers. This provides the ability to monitor and adapt in the off-line mode and also aids the scalability of the system. Figure 6 illustrates this. It can be seen that a significant amount of the server functionality has been replicated on the student's computer, including a web server to provide the thin client to overcome security restrictions. There is a potential problem here as we have no control over the student's environment. Thus the local software must work with any operating system, web browser, etc. In short the local server must be completely cross-platform compatible. The minimum we can impose on the students' system is a Java-enabled web-browser, in which the client user-interface is implemented.

The result is a distributed three-tier, client-server architecture. The user interface, course-delivery logic and data access are separated and the client is now only concerned with the user-interface and connection to the server. Most of the course-delivery logic is moved into the middle layer, where it communicates directly with the database. This thin client architecture provides increased security, flexible and significantly easier maintenance [20]. Our addition to this standard approach is to distribute the server function between the education provider and the network of students. It meets all of the demands of the TILE system. A web browser and Java applet provide the client software.

#### **Network communication protocols and firewalls**

One of the major problems faced in the design of client-server architecture is finding a solution that is compatible with network security measures. This may restrict the choice of network communication protocols. Special attention must be paid to the protocol between the local client and the remote server, as both client and server may sit behind a firewall that will not allow some data packets to pass through. In the design, we can not assume control of the security measures implemented in a particular firewall. The particular issues we face in the TILE system are the loading of an applet from the server to the client and the network or socket connection between the client to server applications. It is this link over which we need to pass presentation data, course structure and synchronisation information.

For a client behind a firewall, the firewall will, in most cases, allow access to the web. This means that they can connect to a server using the HTTP protocol. For a client-server system, three firewall techniques may be used: IP filtering, proxy servers and SOCKS servers. Their impact on the client-server protocol choice has been analysed elsewhere [21] and the conclusion drawn is that for complete generality, the HTTP protocol is the best choice of protocol.

#### **Cross-platform issues**

As already mentioned, the software on the student's computer must be platform independent. We have the choice therefore, to

write the application for each platform supported or alternatively to use Java because of its ability to run anywhere. This provides a powerful reason for developing all software for the student's computer in Java.

For the client, web pages and Java applets provide the logic that is required for the adaptive presentation of the educational material. We have to provide the student with a local server and database, to support the other two TILE access modes: *online with local server and offline with local server* modes. Both server and database must be able to run on any platform. We have also developed the server software in Java. Although there are many database systems available, few meet the dual requirements of cross-platform support and low cost. A native Java database would meet our requirements provided that performance is not an issue. There are several open-source Java database systems available [e.g.22,23]. Performance should not be an issue, as the database will only be supporting a single user, the student.

Although it would seem that this strategy solves all of the cross-platform problems, this is not the case. There are several versions of Java development kits (called JDKs by Sun Microsystems). Macintosh platforms will only support JDK1.1.8 on Mac OS 9 and below. JDK1.2, the current standard, is supported on most other systems, but will only be available from Mac OS X. It is imperative that we support the lowest common denominator in developing the TILE system. We are installing software on the student's computer and must not force the student to upgrade either computer or operating system. Adopting JDK1.1.8 is the only answer.

#### **Java applet security issues**

In general, an applet loaded over the network is prevented from reading and writing files on the client file system, or from accessing a local database system, from making network connections, except to the originating host, or from starting other programs on the client computer. This is because an applet loaded from the network is not trusted [24]. There are two ways for an applet to be trusted by the client system. One way is that the applet is stored on the local disk, the second involves digital signatures. We have written programs to test a Java applet's security restrictions on a range of platforms and browsers. We found that an applet loaded from local disk is only treated as trusted by Appletviewer. Neither Netscape nor Internet Explorer trusts applets stored on the local disk to access the local file system or to open a connection to a local database.

Applet signing is another solution to create trust in an applet[25]. To be trusted, an applet must be signed with an unforgeable digital ID and then the user must state that s/he trusts applets signed with that ID. A digital ID is proof of identity of the person signing that applet. For a browser to understand a signed applet, there must be two digital IDs involved: one used to sign the applet and a second installed in the browser and used to verify the first one's authenticity.

Netscape and Internet Explorer both have different approaches to grant trust to applets[26]. For Netscape, extra code, which uses Netscape's specific Java classes must be added into the applet. Signing methods are also different. For Netscape, the files must be signed with a Netscape Object Signing ID

(creating a "manifest" of the files), and then the files and manifest must be wrapped into a .jar archive. For Explorer, the files must be wrapped into a .cab archive and then signed with a Microsoft Authenticode ID. Authenticode signing and Netscape Object Signing are not supported by all platforms. The ways that Netscape and Internet Explorer deal with expired digital IDs is also different. However, the most important and fatal issue concerning the use of signed applets to ensure security is that not all versions of Netscape and Internet Explorer understand signed applets.

### **Student-side application**

Since neither a Java applet loaded from local disk nor a signed applet can provide a satisfactory solution to the requirements of accessing local files and connecting to the database system, another solution must be found. Because there are no security restrictions on Java applications for JDK1.1, the Java local server application and database can provide all local functionality, providing that we can establish a connection between this Java server and the Java applet. When there is a need to access the local file system, the applet sends a request to the server application and it accesses the database or file system. However, because both applet and application run on different Java Virtual Machines (JVM), there is no way for them to share memory. Again, a client-server approach is the only solution. The Java applet is the client, which opens network connections to the local Java server application, which accepts network requests from that applet. The user interface logic can therefore be separated from the courseware delivery logic, with the user interface provided by the Java applet and all delivery logic by the local Java server application. However, there is still a problem in communication between the Java applet and the local Java server application.

### **Communication between the Java applet and the local server**

Java's security policy constrains an applet to be able to open a network connection only to its originating server and then only by naming the host in exactly the same way as the hostname of the HTML page in which the applet is embedded. This means if the HTML page is loaded from some URL, say: <http://www.nzedsoft.com/applet.html>, then the applet embedded in this page will only be able to open a network connection to the host [www.nzedsoft.com](http://www.nzedsoft.com). Neither the numeric IP address for [www.nzedsoft.com](http://www.nzedsoft.com) nor any shorthand notation for that name will work.

For offline access with a local server, the applet, must be loaded from the local file system. But an applet loaded from `file://URL` is treated as a special case for security reasons. Such an applet is not allowed to open a network connection to "localhost", the local machine IP address or to `file://URL`.

There are two ways to solve this problem and one is the already discarded method of signing the applet. The second method must therefore be implemented, and that is to set up a simple web server on the student's computer and to load the applet from this local web server via a URL address. The function of this web server is only to provide the HTML page with the applet embedded in it, all other pages may be accessed directly from the local disc. Because both the web server and the local Java server application run on the same machine, they share the

same host name. Therefore the applet loaded from a local web server may open a network connection to the local Java server application.

Thus, on the student's computer we must install a system, which comprises a database, a limited web server and the local Java server application. The applet that provides the user interface will be loaded from this local system, when the student is using their own computer, or from the education provider's server, when the student is using an anonymous computer. These components have all been implemented in Java JDK 1.1.8.

### **Software required on the education provider's computers**

On the education provider's computers, for firewall security reasons, all transactions with the student must use the HTTP protocol. The requirements of the education provider's server have therefore been met by adding functionality to a standard web server. In this situation however, performance is very much an issue. Although we have offloaded many of the transactions onto the student's local computer, any remaining transactions, to do with synchronisation and anonymous computer use, must be implemented as efficiently as possible to ensure good scalability. This server-side logic may be implemented using a number of different techniques:

- Common Gateway Interface (CGI);
- Web-server specific Application Programming Interface (API); or
- Java Servlets.

CGI is a very flexible technique and is used in many small-scale applications. It can be used to develop an application on any web server and on any platform, and it is programming language independent. Its disadvantages are bad performance and poor scalability, coupled with potential security holes. The scalability is poor, as each new user request requires a new process to be created on the server computer, and if the number of processes created is very large, then the performance of the system will be very poor.

A web-server specific API is much faster, as the application can be multi-threaded, which means that any number of different users will be managed by just a single process. This approach can also be optimized for the target platform. Its disadvantage however, is its poor portability, as each web server has a different API. For legacy reasons this may not therefore be a sound approach.

Java servlets[27] are also an API approach but they are not web-server specific, provided that the web server supports a servlet API interface. Thus the advantages of a servlet approach are a combination of performance, portability, security and the full access to Java's functionality[26]. From our feasibility comparisons, Java servlets outperform the other two approaches.

Thus on the education provider's computers we install a system that comprises a Java-enabled web server, a Java servlet API and a database server. We have much more control over the choice of platform for the education provider; the computer may even be provided as a part of a total package. Therefore the problems that we have faced in cross-platform portability

on the student's computers will not apply.

### The complete TILE solution

The overall architecture of the TILE education delivery system is shown in Figure 6. It is an extension to the classical three-tier, client-server architecture. Students interact with the system using a standard web browser. The intelligent user interfaces are provided by the Java applet. For offline with local server mode, the delivery logic is provided by the local TILE server application and a local database server provides the data services. For online with no local server mode, the delivery logic is provided in the remote TILE-enhanced Web Server and a remote Central Database Server provides the data services. For online with local server mode, the delivery logic and data services may be provided from either local or remote system, as appropriate.

### 5. CONCLUSIONS

We have shown that despite the profusion of tools available for web-based education, there are still problems that must be solved. We have looked at two broad issues, the development of web-based multimedia, which, we believe, if it is to be taken up seriously must be usable by non-experts. We have described tools that are available and, which are still being developed that solve this issue. The second issue is in providing both scalability and flexibility in web-based delivery. This has been solved using distributed systems techniques.

In looking at this latter issues we have analysed the requirements for a third-generation, technology-integrated learning environment from the perspective of both performance and security as related to the implementation of a general distributed system. We have also analysed the student's requirements in terms of modes of access and flexibility of delivery, which will support a range of good pedagogical practices. Our final constraint was to produce a system that did not force the learner into updating his or her computer to a particular operating system or specification.

The conclusions that we have arrived at are that systems design constraints as well as pedagogical requirements can be satisfied by the architecture that we have proposed in this paper. This architecture does seem complex but the implementation may make use of many existing components. Moreover issues such as installation can be automated and software updates need only be made available to the server.

Data, such as the structure of a course, students' models etc. will be abstracted using SQL databases. This and the delivery logic is stored in one or both of the servers, the one on the education provider's side and/or the one on the student's side. A prototype of this system has been implemented using Linux, Apache and Tomcat on the server side and using a Java applet, server, database and web server on the student's side. Work is continuing with the development of authoring interface tools to develop the course structure within the database and link to various media authoring tools such as AudioGraph.

### 6. ACKNOWLEDGEMENTS

We would like to acknowledge the support for this project from

the New Zealand government's New Economy Research Fund (NERF) under contract MAUX9911.

### 7. REFERENCES

- [1] Paul Maharg (2001) The e-University Project: what's next after U N e x t?, <http://www.ukcle.ac.uk/news/directions9.html>(retrieved on 13/6/0).
- [2] Wayne C. Poncia etc. (1999), Web-based Learning for a Digital Generation , <http://www.nlginc.com/nlglibrary/articles/webbasedlearning.html> (retrieved on 8/2/01).
- [3] Barbara E. Truman (1995), Distance Education in Post Secondary Institutions and Business,University of Central Florida, <http://pegasus.cc.ucf.edu/~btruman/dist-lr.html> (retrieved on 8/2/2001)
- [4] Wayne C. Poncia etc. (1999), Web-based Learning for a Digital Generation , <http://www.nlginc.com/nlglibrary/articles/webbasedlearning.html> (retrieved on. 8/2/2001)
- [5] Features/Tools and Tech Info, Comparison Table for all applications (Jan. 3, 2001), <http://www.ctt.bc.ca/landonline/choices.html> (retrieved on 8/2/2001)
- [6] Web-Based Educational Environments (1998), <http://isis.acomp.usf.edu/Web-Based/support.html> (retrieved on 8/2/2001)
- [7] Tools for Developing Interactive Academic Web Courses, <http://www.umanitoba.ca/ip/tools/courseware/evalmain.html> (retrieved on 8/2/2001)
- [8] Overview of Available Web-based Course Management Systems, <http://www.cs.uml.edu/~heines/gowri/cmslist.html> (retrieved on 8/2/2001)
- [9] Jesshope, C.R. and Shafarenko, A. (1997) Web Based Teaching: a minimalist approach, Proc. Second Australasian Conference on Computer Science Education, ISBN: 0-89791-958-0, pp16-23.
- [10] Jesshope, C., Shafarenko, A and Slusanschi, H. (1998) Low-bandwidth multimedia tools for web-based lecture publishing, IEE Engineering Science and Educational Journal, 7 (4), pp148-154, also published in IEE Computing and Control Engineering Journal, 9 (4), pp156-162 and online at IEE Computing Forum, <http://forum.iee.org.uk/forum/library/>, September 1989.
- [11] R. Gehne and C. R. Jesshope (2000) Tools for the production of small-footprint, low-bandwidth, streaming multi-media for distance education, Proc Lifelong Learning Conference, Central University of Queensland (Brisbane, Australia), ISBN 187 6674 06 7, pp240-244.
- [12] C. R. Jesshope (2000) The use of multi-media in internal and extramural teaching, Proc Lifelong Learning Conference, Central University of Queensland (Brisbane, Australia), ISBN 187 6674 06 7, pp257-262.
- [13] C. R. Jesshope (2000) Using AudioGraph in On-line Teaching, Proc Open Learning Conference, Brisbane, Australia, pp315-320, Learning Network Queensland (Brisbane, Australia).
- [14] Barra, M., Negro, A. and Scarano, V. (1999), When the teacher learns: A Model for Symmetric Adaptivity. In:

- Brusilovsky, P. and De Bra. P. (eds.) Proceedings of Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, Banff, Canada.
- [15] C. R. Jesshope, Integrated tools for on-line education, Proc IWALT 2000, Massey University, New Zealand, IEEE Computer Society ( Los Alamitos CA, USA), 2000,205-208.
- [16] Scourias, J. (1999) Overview of the Global System for Mobile Communications, Retrieved from the web on 1/2/00, <http://ccnga.uwaterloo.ca/~jscouria/GSM/gsmreport.html>
- [17] Roelofs, G (2000) Portable Network Graphics, Retrieved from the web on 1/2/00, <http://www.cdrom.com/pub/png/>
- [18] Nikov A. & Pohl W. ,Combining User and User Modelling for User-Adaptivity Systems. Human Computer Interaction - Ergonomics and User Interfaces ,1999 (Eds. H.-J. Bullinger & J. Ziegler).
- [19] [12] Hoschka P. ,Computers as Assistants: A New Generation of Support Systems.( Lawrence Erlbaum Associates Publishers, Mahwah, NJ, New Jersey, 1996) 336-340 (ISBN 0-8058-3391-9).
- [20] Joseph T. Sinclair and Mark Merkow, Thin Clients Clearly explained (Morgan Kaufman, Academic Press,2000).
- [21] Marco Pistoia et. al. Java 2 Network Security, second edition (Upper Saddle River, N.J. ; London : Prentice Hall,1999).
- [22] InstantDB, InstantDB home page, <http://instantdb.enhydra.org/> (retrieved 6/6/2001)
- [23] Hypersonic SQL, <http://hsqldb.sourceforge.net/> (retrieved on 6/6/ 2001)
- [24] Sun Microsystems (retrieved 6/6/2001) Frequently Asked Questions - Java Security, <http://java.sun.com/sfaq/>
- [25] Daniel Griscom Code Signing for Java Applets, <http://www.suitable.com/CodeSigningOverview.shtml#top> (retrieved 6/6/2001)
- [26] Robin Green Overcoming Java Security Problems on Netscape and IE, <http://redrival.com/greenrd/javasec.html> (retrieved 6/6/2001)
- [27] Sun Microsystems Overview of Servlets, <http://web2.java.sun.com/docs/books/tutorial/servlets/overview/index.html> (retrieved 6/6/2001)