Support Initiative for Multimedia Applications

The provision of CBL material over network information servers

Patrick McAndrew
Institute for Computer-Based Learning
Heriot-Watt University

Abstract

The production Computer Based Learning within the UK Higher Education sector means that there is a need to deliver the material produced. This can be achieved using information servers over wide area networks. This report examines the requirements for serving material and considers if the World-Wide Web offers appropriate tools. The sort of information required on the server is considered with the possible security implications.

The methods by which software can be delivered are considered an initial implementation of a possible tool to interpret delivery instructions is discussed. Other related tools within the community are also identified. A project with particular needs for dissemination is also considered as a pilot study to identify their real requirements and specify the tools they would need.

The recommendations of the report are that the possibilities of the World-Wide Web should be recognised by those with a need to deliver material. If the need is to supply descriptions and packages that can be contained within single files then clients and servers already supply the necessary means. However, the efficient delivery of more complex package does require new tools to be constructed.

This report is produced as part of the Support Initiative for Multimedia Applications funded under the JISC New Technologies Initiative.

Contents

1. Introduction

The Higher Education area within the UK has recently increased its investment in the use of information technology for the provision of teaching and learning resources. Many of these initiatives have concentrated on the production of computer based learning (CBL) material. This has left an area of need to disseminate the material that has been produced.

A separate development in the last two years has been the growth in wide area networks and information services that can be offered over those networks. The Internet is now stated to have in excess of 40million users. Within the UK the decision by the Joint Network committee (now UKERNA) to establish a Janet Internet Protocol Service (JIPS) [1] at the end of 1992 allowed all UK universities to establish access to the full range of Internet facilities. This has increased the range of services that can be operated within the UK and also extended access to services across the world.

Bringing these two areas together it is clear that the use of networks for distribution and management of the learning resources being created within the UK is a good idea. This report examines the requirements for such a service and discusses the technologies that are appropriate for this task and the way in which computer based learning could be placed on network information servers.

2. Requirements for a server for Computer Based Learning

2.1 Traditional approaches

Much CBL material is either not formally distributed or, where it is distributed, floppy disks are still the most common media used. There is however growing recognition of the opportunities for electronic distribution of material that is already in an electronic form.

Using a network to distribute programmes has traditionally depended on file based service in particular ftp (file transfer protocol) has been the usual approach within the Internet community. For an ftp transfer the user needs to know the file name and the site name where the file is being stored. Then a series of commands must be given to transfer the files to the client machine before the installation process takes place. This itself can be made difficult by the need to reduce the number of files stored on the server machine and to hold files in compressed format.

Some of the problems of traditional ftp storage have been addressed by providing central indexes of archive machines (archie) and by designing client programs that allow much easier browsing and downloading of files (for example Fetch, xgetftp). There is also a growing research into how the many large stores of files can be searched and whether search agents can be improved [2].

2.2 CBL collections

For the provision of learning material as distinct from other files there is another approach that should be considered. This is the building of collections in a way to encourage the use of the material contained on them. As an example material should be described, include examples of how it might be used and guide to the installation process. Then access can be given of files for dowloading and installing. While much of this is often offered in README files of ftp servers, the newer formats of server allow this to be presented in an attractive and seamless way. Recognition must also be given to any licensing conditions associated with the software and control held over those permitted to have access.

2.3 Data transfer

CBL material is multimedia in nature, that is it will consist of executables produced in various ways and for various platforms. Within those executables different media types, in particular graphics and increasingly sound and video sequences are likely to have been used. This means that files for CBL will often be fairly large and of different application types.

The centralising of files on servers can lead to bottlenecks in the transfer of data over the networks. This can be limited by local strategies such as mirroring of data to closed sites and caching of data when it is fetched from remote sites. Such automated strategies are not without problems. An example cited in [3] discusses an Australia based server which shows a large range of files on other sites, when these files are accessed they are transferred from the original site, cached at the server and passed on to the requesting client. However the wide range of files apparently available has led to the use of this server by clients all over the world often leading to unnecessary transfer into and out of the Australian network. This example emphasises that caching strategies need to consider how the system will be used.

An alternative to the transfer of programs over the network is to enable the control of local programs from the centralised description. Only if the program was not installed or had changed

form that already placed on the client machine need it be transferred from the server.

2.4 Client and server communication

Control of client programs has been considered within the TLTP funded Interact project [4]. The method adopted in that project is to adapt simulation written for the client machines to respond to signals from network browsing programs (in practice Mosaic [5]) and transfer data from the remote server, via Mosaic, to the simulation. In this way examples of previous data can be shared. This is being extended to allow the state of the simulations to be stored at any point and passed across the network. The Interact Communication Facility (ICF), suggested in this system, also avoids the security problems that would occur if arbitrary shell commands were passed between the server and client [6].

A similar approach, again based of WWW servers and clients, is being investigated within the TLTP BioNet project [7]. There a launching protocol is defined and a Windows based client, W3launch, produced to respond to remote commands from the server. The use of an intermediary
program allows the user of the client machines to ensure that only approved programs are run, a command can only be carried out on the local machine if it has been entered in to the list of permitted commands. Adoption of an approach similar to this but extended to include the installation process should limit the need for programmes to be repeatedly transferred from information servers.

These last two examples recognise the common case where installation of programs happens rarely and is under the control of an administrator. This is an important case but increasingly students are encouraged to explore the opportunities available over networks for themselves. If remote courses are to be made widely available in this way consideration should also be given to less limited modes of access.

2.5 Interaction and feedback

Within Computer Based Learning it is often argued that interaction should be encouraged. This applies also to the serving of CBL. Providing information on a large scale also allows the gathering of feedback data from a large number of users. Information servers now have the technology to allow interactions with users and this would typically be in the form of questionnaires to receive comments. The opportunities should also be explored more fully so that software may be customised more closely to the needs of the user based on an initial question and answer session. At a simple level this might be to ensure that the user is provided with the most appropriate form of the program for their platform. Other possibilities include that the program could be altered to operate at the most suitable level of difficulty and with the data files most relevant to the user's subject area.

The management of any feedback and questions asked about the programs can be made more efficient if the information gathered is ordered to best advantage. The Answer Garden concept [8] considers the use of a database containing previous questions and answers and the associations of experts with each part of the structure. A register of experts will be very attractive
to a growing server of learning material. This would allow managed access between those developing and implementing the CBL material and those using the material.

2.6 Summary of needs for CBL serving

The requirements for CBL serving can be summarised as

- Provisions of descriptions of material
- Known access points for material
- Distributed control of servers
- Access management and control
- Capability to serve large files
- Identification of different file types
- Identification of client platforms
- Methods to limit use of bandwidth
- Searching of server based material
- Communication between servers and clients
- Customisation of software to users needs
- Managed gathering of feedback data

3. The importance of the World-Wide Web

The World-Wide Web (also know as WWW or just the Web) is the provision of independent distributed servers which can work together and link together in a seamless way. The protocol each of these servers use is defined as the HyperText Transfer Protocol (HTTP) to transmit documents in the HyperText Mark-up Language (HTML). HTML allows documents to contain formatting information and links to other media such as graphics, audio and video. The free distribution of descriptions [9], [10] of this mark-up language and information about how to construct servers and clients to display the information has led to a very large number of sites implementing and using this approach.

Public domain implementations of the HTTP servers are available from several sites and for several platforms so that servers can be run from UNIX systems, PCs and Apple Macintoshes. The client software to allow access to server over the network and the display of the retrieved HTML documents is also available for theses platforms and others. In addition for each of the platforms there is a growing choice of client program. This is encouraging for the long term future of the Web approach to providing information, as it seems likely that free clients and servers will continue to be available even as commercial systems start to appear [11].

In addition to supporting documents written in HTML for the WWW clients, the Web offers access to other information services. Allowing access by clients to other servers means that information already being made available on existing gopher and ftp sites can be immediately integrated. A further feature of the Web is that access is possible to any program on the server. This allows easy extension of the WWW system to specialised areas such as database access, on line information gathering and security checking.

3.1 Alternatives to the World Wide Web

The World-Wide Web has now established itself as one of the fastest growing technologies in the computer world. The Web and its protocols and display clients offer advantages over earlier systems such as gopher in the way in which they support structured documents and linking across different servers. However the existing systems does have some weaknesses. In particular communication between the server and the client is not built into the design. Each connection with the server is a new connection and can not be simply matched up to an earlier connection. Software authored for local delivery can also take advantage of additional facilities and produce systems that are easier to use then those currently possible using the web. This may change with later versions of servers, clients and protocols. In addition some alternatives are becoming available, for example the Hyper-G system [12] which offers similar facilities but has additional features to allow greater integration between clients and servers and identification of users.

3.2 Using World-Wide Web for authoring or delivery

The facilities for authoring in the World-Wide Web have not been very sophisticated. HTML documents are basically plain text with mark-up indication (for example <h1> A Heading </h1> is an HTML sequence to define that text as being a level 1 heading). This means that nay editor can be used to create the HTML documents. Tools are now appearing to help with this task, either sets of work processing macros (eg. CU-HTML) or custom programs (e.g. HoTMetaL). There are now a variety of these to help in the authoring task, particularly in the creation of valid HTML documents.

The Web approach to authoring documents is not as sophisticated as that used typical authoring

tools (such as those discussed in [13]). This is partly because of the need for further development of the tools to assist the creation of the documents and also the need to introduce further features into the servers and client, for example tables, pop-up windows and in-line video. Some of these features are planned for future releases such as HTML2 and HTML3 [14]. There is though a fundamental difference between the client-server approach of the Web and most other authoring approaches. On the Web it is under the control of the client how things are expected to appear on the screen. For example one client may choose to render a level 1 header in black bold Helvetica font at size 32 point, while another may render it as blue italic Times-Roman size 12 point. Theses local decisions mean that it is difficult to ensure that a given document appears as intended by the designer, the designer also can not predetermine the delays that will be imposed by using a remote server.

This variation in appearance and interaction response times is not a problem for many uses of the Web where the underlying information is what is important, but where the design has been made to ensure usability these changes can hinder the use of learning material. In practice many Web pages have been designed on assumptions based the use of the most common client with its default settings. Some control can be regained by using active graphic maps [9] in place of text but the current implementation of these does not allow adequate feedback and response times become even more dependent on the network performance.

The majority of courseware has been produced for delivery to local machines using authoring systems targeted at those machines and for the reasons outlined above this is likely to continue to be the case. The role of a courseware server then can be to deliver this material in a simple and flexible manner. As the separate strand of courseware developed for the Web (and similar technologies) develops and becomes established this will be able to be served alongside the more conventional packages.

4. Delivery of courseware

4.1 Sources of courseware within the UK:TLTP, CTI centres and ITTI

The two stage initiative of the Teaching and Learning Technology Programme (TLTP) has led to the funding of over 70 projects at a projected cost over 4 years of approximately £35million. The nature of these projects vary but the majority take the form of consortia involving several universities working within a single subject area. The planned effort of many of these projects has been the production of material.

The Computers in Teaching Initiative (CTI) centres have been established in subject areas to promote the use of Computers within the teaching process. While the approach used has varied between different centres effort has concentrated on the collection of information about existing packages and distribution of that information in booklets and newsletters.

The ITTI projects have in turn concentrated on the training and staff development aspects of the introduction of technology into teaching. The material they have produced is primarily training packages, for example for the Guide authoring program and the use of multimedia in teaching. Recently the central distribution point for ITTI material has established an information server based at one the sites involved [15].

4.2 Dissemination initiatives

The existence of a range of material and the technology to distribute it suggests that the formation of units to disseminate the information should be considered. These need to promote the best practice and material associated with the other initiatives across a range of subjects. An important aspect of the dissemination process will be the integration of information services into their task. Within Scotland there has been a recent decision to establish a centre based on this approach under the Learning Technology Dissemination Initiative (LTDI) [16]. This will make use of WWW technology in its initial service.

4.3 Existing information servers

The growth of information servers in all areas has lead to some attempts around the world to help disseminate learning material. These can be divided into:

traditional ftp server: e.g. src.doc.ic.ac.uk, unix.hensa.ac.uk etc.
self contained systems: Global Network Academy
hybrid systems: using the technology to deliver local clients

It is in this last category that this report has concentrated.

4.4 Dynamic pages and databases

The initial organisation of a CBL collection should be fairly simple in its approach. This suggests that pages should be constructed under the control of an evaluation and dissemination centre based on information provided by the authors. As the Web will be used it is then easy to distribute management of such as server around different sites. When the collection grows in size, however, it becomes appropriate to consider alternative ways to organise the material. An option to be considered is to create some of the page of information to be viewed on a dynamic basis. For example, a database of information about the software could be created and the information needed extracted from this database, a page dynamically created on the server and

then sent and displayed on the client. This offers the potential advantages of a uniform appearance and a searchable database. As a disadvantage some of the flexibility of separately created pages would be lost. In practice the best solution is likely to be a combination of static and dynamic pages with an associated searchable database.

4.5 Security and management issues

Access to remote information and packages raised two security concerns: the need to ensure that only appropriate users have access to the data on the server; the needs as a user to be user that data received from the server will not harm their local system.

In the case of server security this a concern that is being addressed by the community [17]. The initial design of the Web and most other Internet information services is based on public access. The servers are in the main protected against access that is harmful to the serving systems but information can be most easily provided if it is freely and openly available over the network. For the case of courseware material it is likely that some, and maybe most, of the material can only be disseminated if some control is operated over who has access to the material. For example much of the output of the TLTP programme is considered to be available to other UK institutions but not to those abroad. control can be impose based on machine identities or on personal identities. In practice is simple for server to recognise machines (based on IP addresses) but not individual users. For user identification a protocol has been established (ident) but it is not widely used and is not appropriate for client machines where users do not have to login.

In the absence of sufficient remote identification the server itself may need to authenticate users by requesting a user name and password. The continuing identification of the users while they are using the system remains a problem unless dynamic pages are used which carry forward the user's identity. The techniques currently being discussed are unlikely to ensure complete and reliable validation of users, further possibilities are to use a separate exchange of keys (for example a system based on PGP [18]) or examine alternative servers such as Hyper-G [12]. It is important to consider that limiting distribution is only one form of control over use of the software. Important additional factors are enforcing the copyright, using "shareware" licences, and only offering support to those who have registered their use of the software. The logging information provided by electronic distribution could help in many ways with this form of control, which in any case is may more appropriate in the Education sector.

Client security can also be helped by some the techniques mentioned above. The client can ensure that the material is coming from an established server based on machine address and could use the public key approach of PGP to validate any material delivered. Most existing servers of software however rely (fairly successfully) on trust and the users exercising reasonable care. For the service envisaged above if the installation process is to be made more transparent, this may not be secure enough. Greater security can be given if an intermediate client program with limited privileges and capabilities is used to carry out the server instructions, this would be similar to the systems discussed in [4] and [7].

5. Delivery of material from networks

5.1 Approaches considered

In this section practical approaches to the use of the Web for CBL delivery are examined by looking at different possibilities using the technology. The first of these examples is a fairly simplistic approach, which does not satisfy all the desirable security and efficiency aims but serves to illustrate the possible mechanisms available. Next systems that can be found in use among other projects are considered, in particular the BioNet and Interact TLTP projects already mentioned about ([4] and [7]). For further details of these contact can be made with the appropriate projects and authors.

The final case considered is the pilot study construction of a server within the Institute for Computer Based Learning to meet the needs of the Learning Technology Dissemination Initiative (LTDI). This pilot study led to a set of prioritised requirements for a server and the construction of an initial set of tools to start to meet these needs.

5.2 Information required for each item.

If CBL material is to be usefully supplied from a server it needs to be organised and presented in a consistent way. Items that should be supplied could include:

- Identity of the Author
- Ownership of the material (project or institution)
- Date material was created or made available
- A description of the package
- Illustrations from the package
- Requirements for the package
- Access to a demonstration version of a package
- Access to a full version of the package
- Opportunity to give feedback
- Links to further related information or packages
- Examples to be used the packages (where relevant)

These pieces of information are not essential before a package could usefully be placed on the server and distributed. However, it is the lack of this sort of information which inhibits the take up and use of material that is otherwise available for distribution. Some of features are shown in figure 1.

In this example the items shown underlined are exploiting the linking capabilities of the Web to connect to further information stored about the author, the project etc. Selecting the links to the chosen version starts a file transfer to deliver the material to the client machines. Depending on the type of the file this could start running immediately or be installed an executed separately.

This is shown here simple as an example of what is possible. However it would be possible to build up a useful collection based on delivery for local installation with any changes to server or clients.

Using mime types to launch programs

The default action for unrecognised files types in most WWW clients is to transfer the files to the local machine. It is then left to the user to install the package in the appropriate way. This is similar to the process that needs to be undertaken when transferring files from ftp sites. The action that is taken by the client can be altered depending on the mime-type of the file. This is most easily controlled using the suffix on the file. For example the commonly used PC archiver and compression utility PKZIP uses .zip as a file extension. On the server this is mapped to a

mime-type in one of the server configuration files (mime.types for the NCSA server) using a line of the form:

```
application/zip  zip
```

This associates the ending .zip with the mime-type application/zip, the client then associates this mime-type with a program available on the client machine, e.g.

```
application/zip= c:\utils\pkunzip %ls
```

The %s indicates that the file passed form the server should be passed as an argument to the program. (Note the exact form and location of these entries varies with the client or server and the platform).

To deliver CBL material the types can be extended to include suffixed associated with know types of authored CBL. For example CBL produced for PC compatibles is often authored using Authorware Professional or Toolbook. In either case these packages are normally executed using a run-time engine based on the client machine. Under the assumption of local availability of these run-time engines a package could be delivered and launched direct from a page of the World-Wide Web by making the following mime associations:

On the server:

```
application/x-authorware app
application/x-toolbook     tbk
```

On the client

```
application/x-authorware= c:\apw2\runapw2.exe %ls
application/x-toolbook    = c:\toolbook\tbook.exe %ls
```

The use of an x before the mime description signifies an experimental mime type as distinct from those registered and widely used

Limitations of this approach are that the package is always transferred from the server even if transfer has already just taken place, there is a dependency on the run-time components, and new mime types are needed for each different authoring package.

A further problem can also be identified when this is applied to the Apple Macintosh machines. A Macintosh file consists of three parts, the data fork, the resource fork and the file information. When a file is transferred between platforms the special structure is lost unless preserved by encoding the file. As a result any file where the information in the resource fork is needed cannot be associated directly with its run-time component, or immediately executed if it is self contained.

Interpreting instructions from the server

The limitations of associating different file types with their run-time engines one possibility is to use an interpreter on the client to carry out a series of instructions delivered from the server. This can be most easily illustrated with reference to the Unix server and client. One of the default mime types recognised by the Unix WWW server is the shell command interpreter sh.

Files ending .sh are mapped as:

```
application/x-sh sh
```
On the Unix client this is enabled by entering in the .mailcap file
```
application/-sh; sh %s
```
Sequences of shell commands can be stored on the server and executed on the client by selecting the appropriate link. For example (see figure 2) is foo.sh contains
```
echo "Your home directory is  "
cd
pwd
```
this can be reference inside an HTML document as
```
<a href="foo.sh"> select this to see your home directory </a>
```

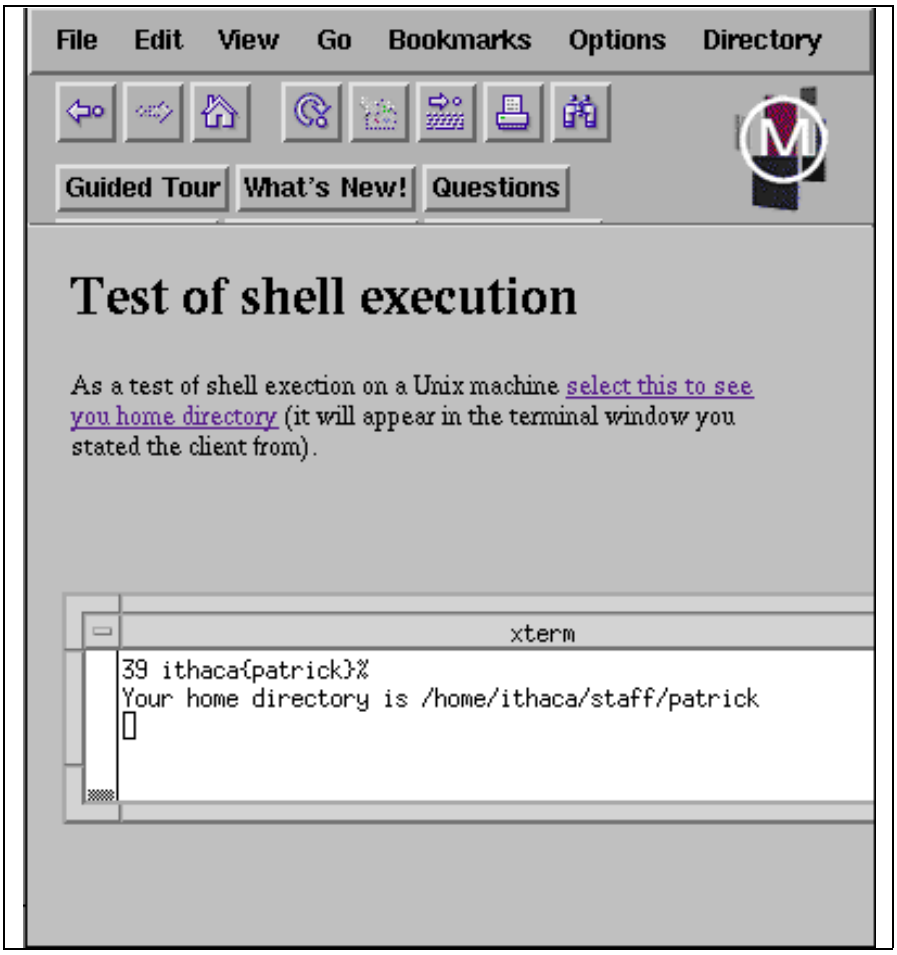

Figure 2: Example using the shell to execute code from the server on the client

Using the shell as interpreter allows all forms of command to be processed including file transfers, copying, deleting, and program launching. This demonstrates that a sufficiently powerful interpreter is available to meet the need fro installation and launching of packages. However it is too insecure, enabling general interpretation of shell scripts would allow, for example, the script
```
cd / ; rm -rf *
```
which attempts to delete all files from the client system!

An interpreter more limited in ability but capable for meeting the requirements for transfer and installation is therefore needed. It is possible that a general interpreter such as safe-tcl a restricted

version of tcl[19], which is deliberately limited in scope may be suitable, but the installation process does involve some files operations which are unlikely to be available in a system that is both general and secure.

An example interpreter

A specialised interpreter can be constructed in any language, for the trail system perl[20] was selected. This is suitable because it possesses powerful text and file handling capabilities and it has an associated library for working with Web services. In particular a "url'get" construct exists in the library allowing a perl script to access any file stored and referenced on the Web.

As proof of concept the initial interpreter can interpret the following information:

```
Package_name
Version
Remote_location
Local_directory
Launch_command
```

An example file on the server might contain:

```
Package_name=Test
Version=1.1
Remote_location=http://www.icbl.hw.ac.uk/patrick/testprog
Remote_location=http://www.icbl.hw.ac.uk/patrick/test.data
Local_directory=testdir
Launch_command=testprog test.data
```

The result of selecting this file is then to install testprog and its data file test.data in a subdirectory of the designated install area and then run the programme on the client machine. The installation can then be recorded locally so that if the same package and version is required a second time only new files form the remote server are transferred and launch command carried out. In particular if exactly the same process is rerum only command script is transferred across the network.

For the example the command files are stored with the suffix .cbl and the a new mime type defined on the server as:

```
application/x-cbldownloader cbl
```

On the client the entry associated with this mime type with the interpreter written in perl:

```
application/x-cbldownloader; dowload.pl %s
```

No further mime types then need to be defined.

The creation of the .cbl structured file can be assisted by using WWW form facility available in most clients. The form show in figure 3 is completed to specify the information required. This form contacts a program running on the server to return the correctly formatted data to be stored in the .cbl file. This ensures that the lines needed in the interpreted file are correctly constructed. This use of forms can also be extended to centrally store the download instructions in a collection for dissemination.

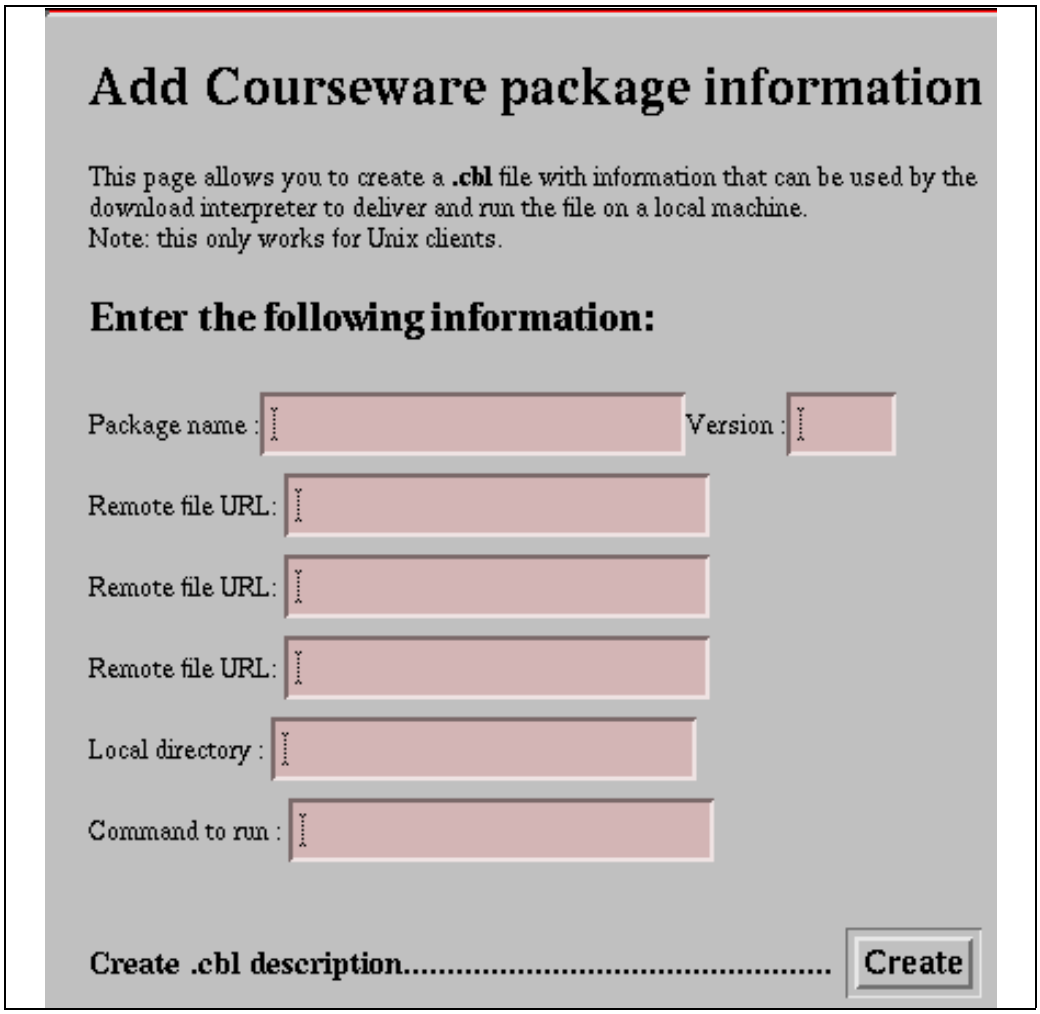

Figure 3: A form to help construct download descriptions

Developing the interpreter for other platforms
This example has shown that with no additional features it is possible to deliver material, however it quickly becomes desirable to have a special downloading process. A possible approach using a download interpreter has been described. The version defined here would need further development to address all security issues and all needs of the download process.

The example interpreter runs successfully on Unix platforms but there are potential problems in generalising it for other platforms. In particular for PC programs this type of interpreter might most simply be constructed as a batch process or in the PC version of perl. However the majority of PC CBL programs run under Windows. This means that it is necessary to launch the Windows program from DOS, this can be done by using client-server programs available for the PC but it does introduce a further level of complexity. An alternative is to program the interpreter to run under Windows by using other development languages.

5.3 W3launch - a World-Wide Web launch program

The BioNet project funded under TLTP has identified and encouraged a wide range of CBL packages related to Biology and Biological Sciences. The main aim of the project has been the introduction of these into teaching at many Universities. As a part of this the project has made use of WWW and ftp server to disseminate the programs.

Some of the packages have description pages set up as "student pages" as in figure 4. From these pages students can make selections and run programs already installed on their local machines. In this example the description includes text and still graphics, and if the student is given the option of running an animation if it has already been installed on the client machine.

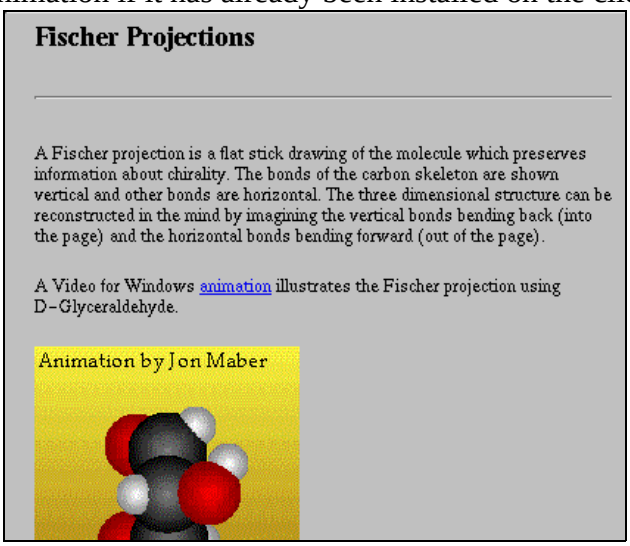

Figure 4: Example student page for use with W3launch

The W3launch program created by John Maber [7] acts as the interpreter to establish which program should be run on the client. The interpreter uses the suffix .w31 and the mime type application/x-w3launch to communicate from the server to the client.

A version of the program is available for Unix machines and for PC compatibles. The PC version has been developed more fully and allows local editing of the behaviour in addition it is constructed to run under Windows to allow the easy launch of other Windows programs.

This package does not attempt to automate the installation process, so avoiding the security problems inherent in transferring programs, but has developed a mechanism to allow the use of programs described on remote Web servers.

5.4 The Interact Communication Facility

The Interact TLTP consortium is developing simulation software for use by those teaching Engineering subjects. As a design decision the target platform is Unix workstations running the X Windows system. This decision was made to provide a platform powerful enough for the simulations to run smoothly, however it also means that the developers have been able to take advantage of the multi-tasking and communication facilities within Unix.

One aim of the Interact consortium was the provision of on line guidance to users. This was achieved by writing hypertext delivered via WWW and Mosaic. However it then became necessary to integrate the tutorial descriptions with the simulation software. For this reason a means of communication from the information presented on the Web and the simulation programs was developed. The resulting Interact Communication Facility (ICF) provides a language to communicate from the server with the simulation programs running on the client machines. The project was not seeking to automate the install process but as both the client simulations were being constructed by the same team they were able to add two way

communication.

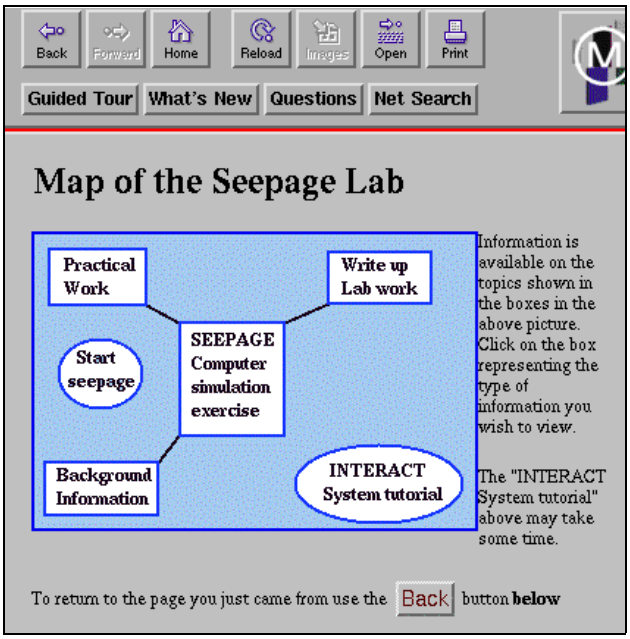

Figure 5: Astudent page for use with Interact simulations

The programs using ICF can not only be launched by selecting item on the Web but also controlled once they are running. This allows the programs to be placed in particular example states (see figure) and then the student can observe and report on the results of the simulation. In addition the specially written simulations can save their states in a way that can be passed back over the web. This encourages the sharing of results and makes it easy for a tutor to save interesting states in the simulations. The Interact team plan to take further advantage of this facility by developing a version the Answer Garden [8] concept so that when questions are asked the answer can include the necessary state for the student to see a particular point the tutor might wish to make.

5.5 Establishing a courseware server for LTDI - a pilot study

Background to the LTDI project

The Learning Technology Dissemination Initiative is a programme supported by the Scottish Higher Education Funding Council with the aim of encouraging take up of computer based learning methods within Scottish Higher Education Institutions. To achieve this the LTDI team is organising workshops around Scotland and arranging visits by members of the team to individuals and Departments to demonstrate and promote the possibilities in particular subject areas.

As a part of the programme a collections of CBL material is being gathered from various sources both commercial and from other initiatives such as those funded under TLTP. This collections needs to be organised so that those on the LTDI team can easily select items for demonstrations.

Within LTDI the World Wide Web is being used to promote the workshops and provide information about the project. The existing use of the Web and the large collection of software were reasons fro selecting LTDI as a pilot for some of the ideas and requirements outlined in the earlier sections of this report. In collaboration with members of the LTDI team specific requirements for their needs were identified and an initial set of tools constructed to meet some

their needs. This work is continuing and it is intended that the majority of the collection will be organised and distributed for demonstration using the World-Wide Web.

Requirements

The LTDI programme has the specific need to demonstrate a wide range of software through workshops and visits by team members. The requirements that they have in particular were established through an interview and discussion about the possibilities. The requirement list produced is given in table 1 below

	Requirement	**Priority**
1	To be able to put demonstration machines into a known state	High
2	Access to all types of PC software (Windows, DOS run-time dependent)	High
3	Ability to remove previously installed software	High
4	Integration of downloads with a database of software	High
5	Ease of access to place material on the server	High
6	No unnecessary transfers of previously installed components	High
7	Customisation of installs to allow partial installation of multi-part packages	High
8	Security	High
9	Access to software for non-PC platforms (Macintoshes and Unix)	Medium
10	Demonstrations available over the web without local installation	Medium
11	Links and compatibility with other sites addressing software distribution	Medium
12	Access to information and programs from CDRoms	Medium
13	Running and customisation of programs by selection from the server	Medium
14	Comments and illustrations on the programs available	Medium
15	Browse software to download to hard disk	Low
16	Easy contact with the maintainers and authors	Low

Table 1: Requirement for the LTDI courseware server

Meeting the requirements

The requirements specified for the LTDI server vary in importance and in the difficulty in which they could be met using a remote server. The overall requirements from LTDI also reflect the need for installation to demonstrate to others rather than to deliver to the whole community over the net. In this case the users of the dissemination system are likely to limited initially to members of the LTDI team and the server is constructed to not be visible outside the host institution.

Some of the requirements, for example to return to a known state, to be able to remove software and to be able to avoid unnecessary downloading of code, imply that a record needs to be kept of the state of the machine that is receiving the software. The need to browse and select software

and to vary the configuration of packages before they are downloaded, imply interaction with the server and querying of data stored about each program. As the programs will not have been constructed with this form of installation in mind the information needed for installation remotely also needs to be gathered.

At this stage there is no requirement for wide accessibility so that solutions based on local area network technology could be considered, however much of the software gathered can be freely redistributed so that a wide area distribution will be required in the future. The World Wide Web technology therefore could offer a solution to current problems and in the future allow much wider access. The initial target machines are PC compatibles reflecting the greater volume of software being produced for those platforms and also the greater need for special installation procedures, particularly for programs running under Microsoft Windows.

The solution suggested consists of the following elements
1. An installation monitoring program: This is used by the person who is to install software onto the server. As a first stage the software is installed on to one of the target machines and the monitoring program reports all changes caused by the installation. This record can then be used for subsequent installations and for returning the machine to the state it was in before the software was installed.

2. An installation database: This records the packages installed and any dependencies between the package and other programs or run time routines

3. An installer client: This receives information from the server and interprets the instructions in that information to fetch and install the parts of the software.

4. Installation pages: Based on the server these pages describe the software and how it can be transferred to the client

5. Server utilities: On the server there needs to be a database of the software and background information such as contact details.

The installation monitor uses a stored state of the target machine and a perl script to do a comparison of all files likely to be changed. This could be all files and directories on the machine but in practice it is found that the time consuming comparison of all files can be replaced by an analysis of those in particular directories only. For example, Windows files need to be monitored but not those related to networks or the basic DOS operating system. The different files installed are stored in a list indicating if a new file is installed or if changes have occurred in text files (such as WIN.INI) the exact lines changed are recorded.

The installer use the files installed during the monitored installation process and the comparison of states to automate the installation. In this case the installer instructions are passed to an interpreter on the client which copies the new files and makes any necessary changes to existing files. For the pilot version of this tool the transfer is made using the local network (PCNFS) rather than the Web. It would be possible to construct a system similar to that designed around the url'get construct described above. However this would need the appropriate function to be created for the PC platform and is not required to meet the direct requirements for the LTDI pilot. This is however a very important extension if the program can be used in more general circumstances.

The installation pages are constructed form the monitored installation process. Theses need to be developed to allow the user to specify the part of the material to download. The Form interface within the WWW clients is a suitable way for this to be specified, the download script then being tailored to the specifications given by the user. An existing database of software can also be transferred across for delivery over the Web by building pages dynamically from the information in the database, this information then integrating with the download process.

Further work

The software to meet the requirements of the LTDI team is under development. The experience of the LTDI team has revealed that installation and removal of software is not a straightforward process, particularly when under MS Windows. The automation of this process by monitoring and recording of actions should enable subsequent installations to proceed more easily. Theses tools will initially help with the management of the local area network but by building them in conjunction with the technology of the Web they can be extended to allow complex installations to take place over wide area networks.

6. Conclusions and Recommendations

t seems clear that information servers will play an increasing role in the distribution and promotion of courseware material. This study has identified some initial requirements for the use of such servers. Those requirements can be partly satisfied by the facilities currently offered by World-Wide web servers and clients. In particular advantage can be taken of the mime facility to extend the recognition of special files to include pieces of CBL that can be contained in a single file. In practice the limitation of this approach is seen when applied to existing packages (especially those designed to run under MS Windows). It then becomes necessary to construct a program to be based on the client to interpret instructions delivered from the server.

An example interpreter constructed to run on Unix clients has demonstrated the feasibility of the approach, however development would be needed if the interpreter was to be used on other platforms. Other projects have also considered the need for communication from information contained in Web pages and programs on the client machines, though without attempting to integrate the delivery and installation process with the running of the programs. The ICF developed as part of the Interact project demonstrates the potential for communication in both directions and has also demonstrated that the hypermedia facilities of the web provide a good way to describe tutorial material, particularly when examples and simulations can integrated with the hypertext. The BioNet consortium have also developed a useful launch tool W3launch that works well with MS Windows applications.

The requirements of the LTDI project where dissemination and demonstration of packages is part of their aim has revealed both the need for a network based delivery system and the complexities that occur in many of the installations. The initial version of tools to meet those requirements has meant devising a monitor for the installation process and a tool to replay and undo those installations. The configuration process uses the web while transfer uses the local area network. This restriction can however be removed to allow the installations to be delivered over the Web.

Summary

- The use of the World-Wide Web to publicise and provide access to material is simple and effective.
- The automatic installation of programs is less simple and raises issues of security and network loads
- For general installation of software an interpreter program for client machines needs to be developed
- The automatic running of previously installed programs has less problems and solutions are being developed
- Installation of software is a problem are on local as well as wide area networks and there is a need for the solutions

7. Acknowledgements

I would like to acknowledge the assistance of my colleagues within the Institute for Computer Based Learning at Heriot-Watt University, especially Richard Kirby from the ELF project and Rupert Loader from the LTDI team. I would also like to thank Jon Maber (BioNet) and Calum Smeaton and Alan Slater (Interact).

8. References

[1] Day, R.A., "Overview of the JANET IP Service and of Requirements for Connection", Joint Network Team, January 1992.

[2] Kirby, R., P. McAndrew, A. Kilgour, H. Taylor, and T. Mayes, "ELF: The Electronic Learning Facilitator", to appear in proceeding of ALT-C, Hull, September 1994.

[3] Schwartz, M.F., "Internet Resource Discovery at the University of Colorado", IEEE Computer, September 1993.

[4] Slater, A.F. "Controlled by the Web", First International Conference on the World-Wide Web, pp 399-409, Geneva, Switzerland, 1994

[5] NCSA Mosaic Project, NCSA Mosaic Home Page, http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html

[6] NCSA Mosaic Project, NCSA Mosaic for X Documentation, http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Docs/mosaic-docs.html

[7] Maber, J., W3Launch - Secure Software Menus on the World, http://www.leeds.ac.uk/bionet/w3launch/w3l-home.htm

[8] Ackerman, M.S, and T.W. Malone, "Answer Garden: A Tool for Growing Organizational Memory", Conference on Office Information Systems '90, pp31-39. ACM Press, 1990.

[9] NCSA Mosaic Project, World Wide Web, http://www.ncsa.uiuc.edu/General/Internet/www-overview.html

[10] CERN WWW Project, World-Wide Web Home, http://info.cern.ch/

[11] Netscape Communications Corporation, Welcome to Netscape!, http://home.mcom.com/home/welcome.html

[12] Kappe, F., and H. Maurer, "Hyper-G: A Large universal Hypermedia Syatem and Some Spin-Offs", ACM Computer Graphics, May 1993.

[13] Riley, F, "Understanding IT: A Review of Hypermedia Authoring Packages", USDU, Sheffield, 1994

[14] Raggett, D., "A Review of the HTML+ Document Format", Proceedings First International Conference on the World-Wide Web, pp5-14, Geneva, Switzerland, 1994

[15] ITTI, ITTI Home Page, http://www.hull.ac.uk/Hull/ITTI/itti.html

[16] Rist, R., Learning Technology Dissemination Initiative, http://www.icbl.hw.ac.uk/projects/LTDI.html

[17] Freeman-Benson, B.N., "Using the Web to Provide Private Information", First International Conference on the World-Wide Web, Geneva, Switzerland, 1994

[18] Zimmerman, P., et al. "Pretty Good Privacy: Public Key Encription for the Masses", ftp://unix.hensa.ac.uk/pub/uunet/security/virus/crypt/pgp, July 1993.

[19] Ousterhout, J.K. "Tcl and the Tk Toolkit", Addison Wesley, 1994

[20] Wall, L. and R.L. Schwartz, "Programming perl", O'Reilly & Associates, 1991