5.2
The Electronic Hartlib Project

Edward Ng, Stefano Mori

School of Architectural Studies
Sheffield University
PO Box 595, The Arts Tower, Sheffield S10 2UJ, UK

One of the many criticisms of early efforts in multimedia based teaching, learning and information systems is that most of the development is focused on constructing closed systems, and that once they are completed, altering their content, especially by third party users, is next to impossible. This leads to two problems. Firstly, in the current funding environment, it is almost impossible to sustain the system. Secondly, the system thereby developed is not very flexible and hence can be difficult to use. In Sheffield, we are trying to address this problem by constructing an open system. Using an interface-less data structuring system, an object oriented technique has been developed to separate the interface from the generic files thereby allowing unlimited posthumous alteration and adaptation. A prototype has been developed in Hypercard and in Director, but the beauty of the system is that it can be adapted to run on almost anything.

Introduction

Fuelled by the availability of cheaper and more powerful hardware on the one hand, faster networks and bigger, more reliable storage media on the other, multi-media - an invention of the 70s, has recently become a possible alternative to traditional classroom style teaching. There are a number of ready made software packages which are geared to provideing authoring environments for those brave enough to venture into the 'unknown world'. Director, HyperCard and SuperCard on Apple Macs, and Guide and Toolbox on the PCs are just some of the ever increasing array of software available on the market. In addition to platform specific software, cross platform and more powerful software is also now available for larger networks made up of various operating systems. Authorware Professional is just one such example.

The problem with most of these software packages is that they are not designed for the light-hearted. Unless one is very good at authoring and scripting, or has the help of an experienced programmer, working with any of this software is never easy. For example, in Sheffield, only two out of 23 staff are multimedia alert, 5 more can be trained to take it up without much difficulty, but the rest simply have no clue what so ever. All they know is word processing with a bit of drawing attached to it. That is to say, on average, 70% of all teaching staff are either not interest in multimedia, know that it is important but do not want to spend any time learning it, or simply are scared by what is involved.

The second problem with most of the software, with the possible exception of Authorware Professional, is that most of them are 'closed systems', that is to say, once they are completed, altering

their content, especially by third party users, is next to impossible. This is mainly due to the fact that the content of the lecture, the material they used and their organisation is closely linked to the interface, the structural and the scripting environment. For example, when Hypercard first came out some seven years ago, it was heralded as the authoring tool for the rest of us. A lot of effort has been spent making it more available and useful since then. A huge number of stacks have been created, some more useful than others. However, although making a simple stack from fresh is simple enough for some of us, combining two or three useful stacks into one is down right difficult even for the most experienced.

In short, the two most daunting questions facing the future of multi-media for the 'rest of them' are still accessibility and flexibility.

The Classroom as a Metaphor

What happen in a traditional classroom? Lecturer A comes to the lecture theatre with his slide cassettes. He loads them onto the projector. He distributes some handouts if any. He waits until everybody is settled down and ready. He then fires away with slides which he organised the day before the lecture. While showing the slides, he makes comments on each one of them, makes some important points, supplementing them with simple drawings and annotations on the overhead projector. At the end of the lecture, he will ask if there are any questions. He may even set the students some simple tests to make sure they understand the lecture. There are a number of things that need to be noted about the whole sequence. Firstly, how lecturer A organises his lecture. Secondly, how he presents the lecture. Thirdly, how he makes sure he is doing his job properly.

Now imagine what happens when lecturer A reaches the age of 65 and is going to retire soon and lecturer B has been asked to pick up the lectures. Lecturer B would probably inherit a lot of the material lecturer A used in his lectures. Some of this material is likely to be out of date while some still useful. Lecturer B probably has material of his own. He probably has his own way of teaching, his preferences and his interests. He is likely to alter lecturer A's lectures. He may adopt what is there and modify it to suit his needs. He may draft out a completely new structure and simply use some of the old material when he sees fit. He may use the old material in a completely new way. He may even retouch it to stress a slightly different point. There are now a number of things that need to be noted. Firstly, how lecturer A is going to explain to lecturer B about all the 'old material' and the way it is organised. Secondly, how lecturer B is going to comprehend and make sense of lecturer A's lecture. Thirdly, how lecturer B is going to change lecturer A's lectures to suit his own needs.

These are the real issues and sometimes, overwhelmed by the latest technology, we tend to forget about them. We tend to believe that the rest of them should learn how to use the computer and that they should work the way the computer works. Not so, I am sure that if we cannot bring computers that one step closer to what people are accustomed to doing, in ten years time, there will still be 10% of us wondering why the 90% of the rest of them cannot be like us.

The CDF Project

The CDF (Curriculum Development Fund) project is part of the Sheffield University IT initiative to develop systems on computers for a more student centred learning environment. The project started in July 1993 and will end in September 1994. CDF was initially set out to fulfil two objectives. Firstly, to develop a system for the majority of the teaching staff in the University, the 90% of the rest of them, which is easy to use and simple to navigate. Secondly, to develop a system which is flexible, expandable and adaptable.

After some prolonged discussion with a small group of 'the rest of them', an initial project specification was drafted. The idea is very simple. Firstly, for the rest of them, they should be able to author a multimedia lecture with no more knowledge than using a simple word processing package with a bit of drawing attached to it. Moreover, for the rest of them, they should be able to change any existing multimedia lectures with no more knowledge than editing a piece of writing on screen. Secondly, for the rest of us, we should be able to update the package every now and then to cope with any newer and better technology without harming the content of the lectures. Last but not least, the users should be able to customise the look and feel of the lecture to suit their particular preferences and styles of learning.

What we are proposing is a three layered structure where each layer is an independent entity in its own right but could be linked together to form a coherent whole. The three layers are Generic Files, Sequencing Files and Interfacing files (Figure 1). This object oriented approach enables us to create a much more open and adaptable system which caters for the needs of the various parties at different levels.

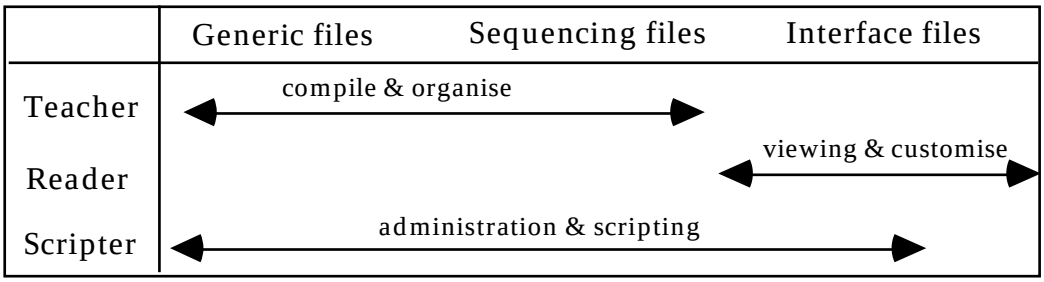

Figure 1. The three layered structure of CDF

Foremost of the three tiered structure is the Generic Files. These are the meat of the lectures. At present six different types of generic files are supported in the current version. They are text, pictures, sound, animation, video based movie and voice synthesised text files (Figure 2). More file types will be supported in further versions of CDF. At present, most of these file types only work on Apple Macs. However as CDF develops, appended lists can be added to enable cross platform compatibility.

TextASCII files
Picture8 bit PICT files, 320 x 240 pixels in size
SoundAIFF files, no compression, 8 bit mono.
AnimationPICS files, various frame rate, 8 bit, 320 x 240 pixels in size
VideoQuickTime files, 320 x 240 pixels in size
VoiceASCII files, Apple Plaintalk Text to Speech conversion.

Figure 2. Specification of the Generic files

Next in the structure is the Interfacing file. The interface file is the engine of the lecture. It contains all the code required to run the presentation, interact with the user and load the right file at the right time (Figure 3). At present, the CDF team has completed scripting on HyperCard and on Marcomind Director. That is to say, the same lecture can be played using Hypercard or using Director (Figure 4). The look and feel of the two interfaces can be very different but the content remains the same. The interface file is meant for the 10% of us who are fluent with scripting on computer.

```
on loadSequenceFile seqFile
global sequenceList, sequenceKey, maxList, maxRecord, rootPath, dataPath
put "" into sequenceList
put rootPath & seqFile into filename
open file filename
repeat
read from file filename for 16384
if it is empty then exit repeat
put it after sequenceList
end repeat
close file filename
```

Figure 3. Some sample interfacing code

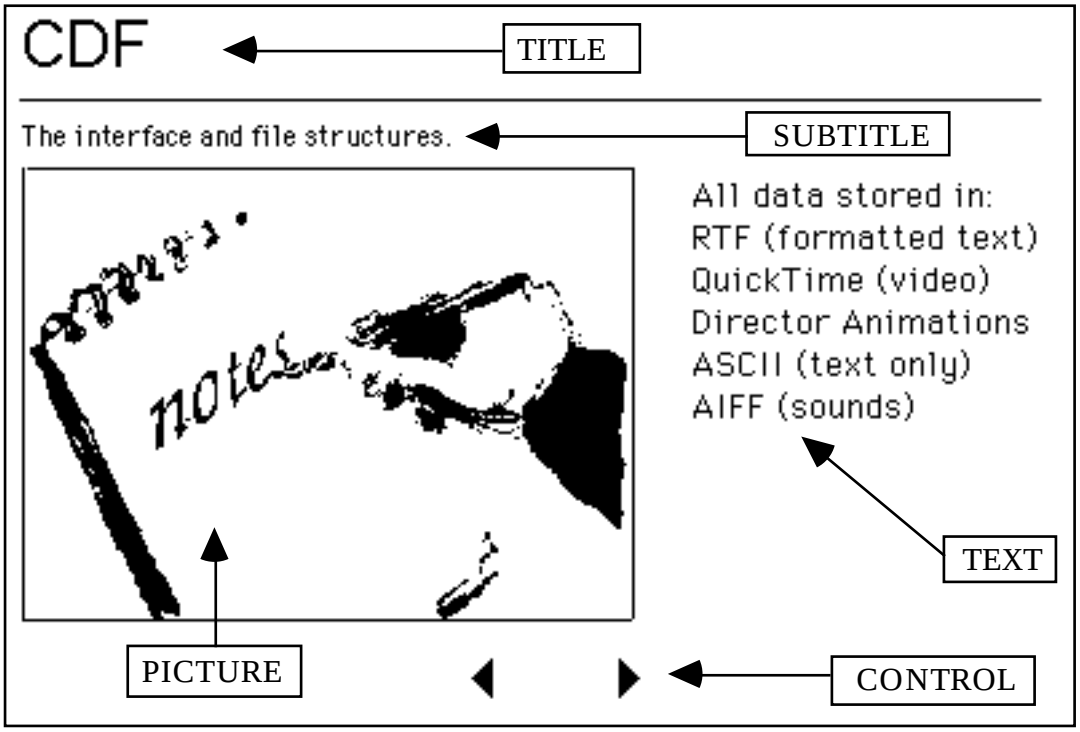

Figure 4. An interface on Hypercard

Controlling the whole thing is the Sequence File. It is the skeleton of the lecture (Figure 5). The Sequence File is basically an ASCII text file. It is used to instruct the interface what, where, when and in what order the Generic Files should be shown. The Sequence File can be created and edited using any word processing program. This File is the lecturer's main point of contact with the system. Since the Sequence File is no more than just a text file, it can be authored and changed easily with a word processing program. An example of the Sequence File and what it means is provided as follows:

CDF SEQUENCE FILE 001	Tell the interface it is one of the many sequence files available
OPTIONS SELFRUNNING	Tell the interface in which mode the lecture should be viewed
CARD	Tell the interface this is the start of a new screen
TITLE Construction	Tell the interface to display 'Construction' as the main title
SUBTITLE Foundation	Tell the interface to display 'Foundation' as its sub-title
ASCII T001	Tell the interface to load a text file call T001 and dispaly it
PICT P001	Tell the interface to load a picture file call P001 and display it
AIFF A001	Tell the interface to load a sound file call A001 and play it
CARD	Tell the interface this is the start of the next screen

Figure 5. The functioning of the Sequence file

What happens at the end is that the user will select the interface file by double clicking it (Figure 6). He will be presented with a list of Sequence Files. The required lecture will start automatically by double clicking the list. What the interface file does is to read the Sequence file line by line, load and unload the required text file, play and quit the required sound, animation and video files and at the end of the lecture return the user to the desktop.

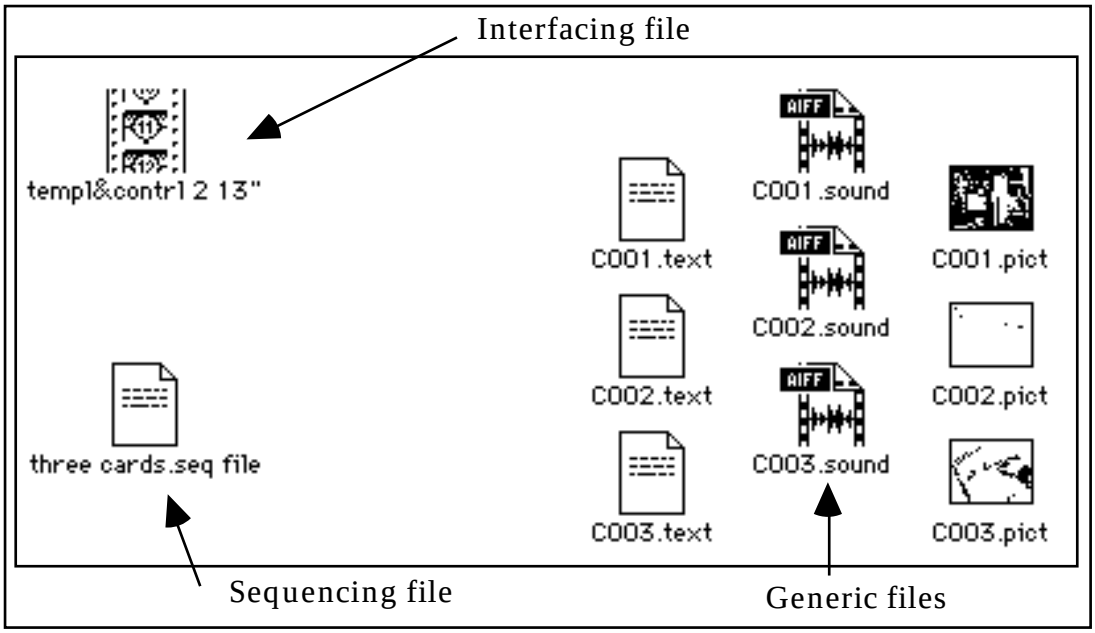

Figure 6. CDF on the computer desktop

At present, CDF is still undergoing some fine tuning adjustment before it can be fully commissioned. Five CAD lectures have now been authored using the beta version of the interface. What is interesting to note here is that, unlike traditional multimedia projects, one does not need to have the final interface all singing and dancing before filling it with the lecture material.

The Future

When CDF was first conceived, a wider picture involving knowledge representation and structuring was proposed. However, as the project digs down into the fundamentals of the implementation of the technology, it becomes clear that the original goal was far more ambiguous than expected within the time frame of the project. What we can talk about at present is the theory behind the knowledge representation aspect of CDF. There are, unfortunately, no demos and no solid implementation yet.

One of the main considerations during the initial stage of formulating CDF was how it can be made more intelligent - intelligent as far as presentation is concerned. Using Professor Lawson's words in one of our internal discussion documents,

"A proposal has been made to include two Prolog Engines into the system acting as filters to the information structure (Figure 7). The proposed system operates by establishing a semantic network of knowledge consisting of three types of elements which are 'resources', 'topics' and 'conceptual links'. 'Resources' are pieces of multi-media data including text, images, sounds and films. A 'topic' is the smallest chunk of knowledge which is presented in an undivided way to the designer. In reality it consists of nothing more than instructions to present one or more resources according to a specific format. Of course, some 'resources' may be used for more than one 'topic'. Especially in the case of films and images. A 'conceptual link' consists of a way of relating two or more 'topics'. Thus one topic may be 'an-example-of' another, or possibly 'a-detailed-explanation-of' another, or 'an-attribute-of' another, and so on.

The Prolog A engine will build a complete tutorial for a designer in response to a specific enquiry by 'tearing' away a fragment of the concept network of knowledge. This is achieved by using predicates to select material appropriate to the enquiry itself and any other supporting material necessary to explain that material to that particular designer. The predicates will support a variety of modes, such as

'teaching', 'browsing', and 'designing', to enable the designer to make enquiries in a form suited to the problem context. Modes will differentially allow designer control of interaction.

The 'designer interface' will actually present the tutorial, allowing the designer to control and manipulate this presentation according to situation and learning style. In addition the 'designer model' held by the system of that designer's knowledge and preferred learning styles will be updated. Here designers will also be able to express learning styles and levels of expertise which inform the model initially.

The Prolog B engine will enable technical authors to express knowledge to the system and to edit the knowledge already in the system. This system will need to allow for the input, cataloguing and reviewing of multi-media resources, for the definition of topics as formatted collections of those resources, and for the expression of conceptual links.

Practically, the system cannot be realised unless it is possible to provide an environment in which technical authors can express their knowledge to the system, both in terms of topics and their conceptual links. Such an environment must not require the technical author to have expert knowledge of the tutorial building software and will require very high level interface design. It must also allow for the knowledge base to be dynamic, with addition, deletion and editing of resources, topics and conceptual links."

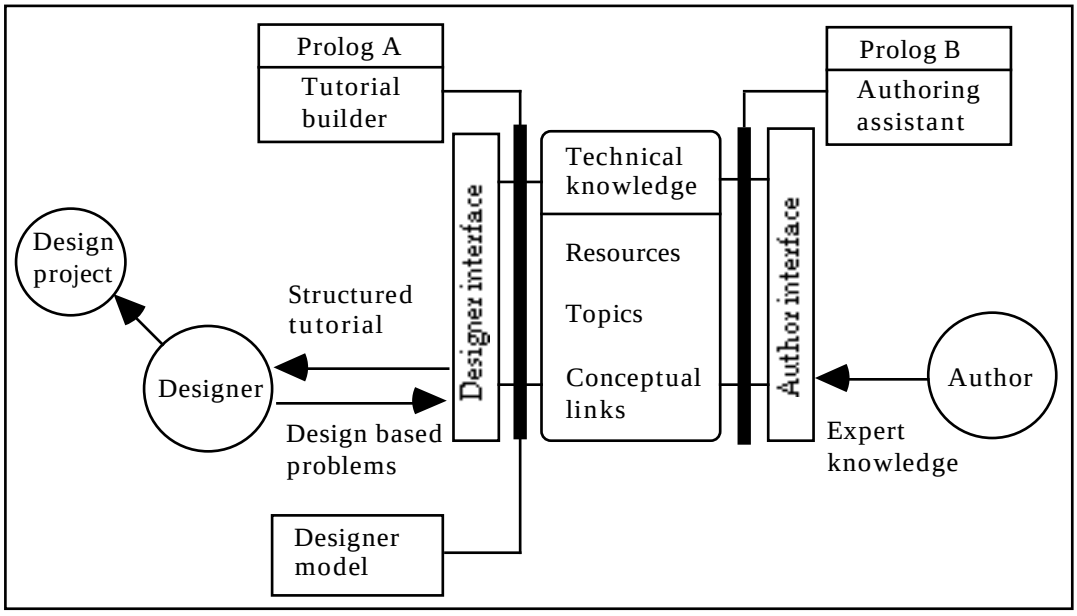

Figure 7. CDF as part of the Knowledge Representation Structure

Suffice it to say at this point that the task is by means a simple one. Hopefully, in parallel to an attempt to improve and publicise CDF as it is, man power can be allocated to its development in the near future.

A Proposal for the Electronic Hartlib Project

Not withstanding the need to take CDF to another level of development on knowledge representation, one of the more pressing consideration now is 'availability'. At present a lot of time is needed to compile the generic files before the actual sequence file can be written. Initial studies indicate a ratio of about 15 to 1 between the time required to collect and compile the generic files and that needed to write the sequencing file. This is highly unsatisfactory and two proposals have been formulated. Firstly, a small piece of software is being worked on to simplify the material compiling stage of the project. This will facilitate

faster structuring of aquired material. Secondly, a call should be issued for a wide collaboration between Universities in Europe to assemble a rich database of Generic files .

It is realised that Sheffield on its own will not be able to accomplish the task at hand without the help and support of our colleagues in Europe, and example is taken from Samuel Hartlib, who's work of almost 300 years ago indicates a good model. Hartlib was a publisher based in London in the middle of the 17th century. Because of the nature of his business, he was fortunate enough to have access to most of the intellectual papers of the time. What he did was to collate the papers that came to his attention and redistribute them to people who were likely to find them useful. A large interdisciplinary database grew to the advantage of the larger intellectual community. In short, the 17th century Hartlib Collection, now archived in Sheffield, is a record of an exchange of knowledge that sought to unite the academic disciplines and complete our understanding of the 'universe'.

Hence we make this proposal: you send us what you can contribute to the CDF project, and we will send you all the material we have including all contributions made by other people. At the end of the day, you will have a generic file collection as rich as everyone else. You can then write your own sequence files and interface files in whatever way you deem best. You could also make use of somebody else's files if you are feeling lazy. CDF allows people who are not multi-media developers to have a substantial part in it's development and benefits. A historian, for example, may contribute a some images, some sounds and a sequence file of his lecture. A programmer may contribute by writing a few lines of codes which will allow us to import other file types or to make the interface a bit more attractive. A student may contribute by mocking up a more friendly and colourful front end. The possibility is limitless.

However, before we can even start working on that basis and to facilitate a wider and longer term collaboration among researchers in Europe, we need to agree on a few things first. The following calls are made.

Call 1 The setting up of a ECAADE sub-committee to co-ordinate future actions.
Call 2 The co-ordination of file specification and technical standards.
Call 3 The development of a mechanism whereby efforts can be pooled and shared.
Call 4 The formulation of publication time tables on a European-wide basis.

Individual invitations to members of ECAADE will be issued after the conference. It is hoped that by the end of this year sufficient interest can be pooled to give the project a head start.

Acknowledgement

CDF is supported by research awards from the ARCUK, the RIBA, University of Sheffield Curriculum Development Fund no. 312 and the University of Sheffield Enterprise Unit. The research team includes Professor Bryan Lawson, Dr Edward Ng, Ms Melanie Richardson, Mr Stefano Mori and Mr Joules Alexandrou.

Order a complete set of
eCAADe Proceedings (1983 - 2000)
on CD-Rom!

Further information:
http://www.ecaade.org