An Intelligent Tutoring System for Teaching CAD

Professor Bryan Lawson and Dr. Peter Scott

Department of Architecture
University of Sheffield
Sheffield S10 2TN
United Kingdom

Abstract
The paper raises some general problems concerning the teaching of CAAD both in schools of architecture and in practice. A new, less 'system-oriented' approach is suggested and some ideas for the design of a computer based intelligent tutoring system are advanced. Some prototype elements of the tutor will be described.
Introduction

In this paper we aim to do two things. Firstly we shall try to raise some general problems about teaching CAAD, some of which are organisational, some financial and others educational. Secondly, and in the light of this analysis, we shall briefly describe some of the ideas and approaches which form the basis of a new research programme begun within the last year at Sheffield University to study these problems.

The organisers of this conference are to be congratulated for focussing our attention, as they did in the call for papers, on the problem of teaching CAAD. Those of us interested in CAAD tend to be both compulsive inventors and designers, and also well and truly bitten by the bug of computing. We generally consider ourselves creative and are most happy when inventing and making things. It is not surprising therefore that we can easily be blinded by our own enthusiasm to the deficiencies of the programs we write and the real difficulties faced by others who try to use them. Thus good empirical studies evaluating the use of CAAD and well developed approaches to training are rare. The amateur CAD enthusiast is notoriously bad at program documentation, and even worse at writing user manuals.

Such work is on the whole tedious and may not seem as exciting as designing and writing CAD programs, but in its way it is just as important. We should never forget the 'aided' in CAAD, which implies that a man-machine system is in operation. The extent to which a designer using a CAAD program understands the program and is a master of it will have a significant influence on the effectiveness of that man-machine system. Today's CAAD systems are sophisticated pieces of software, but they are useless without their equally sophisticated users. Experiences of many CAAD installations employing a wide range of hardware and software reveals that what distinguishes the best from the least successful operations is much more often than not factors to do with the users rather than the CAAD system. In fact this experiential and anecdotal evidence suggests that an estimate as high as 80% user factors might be reasonable.

It is hard to get any accurate data on the real usage of sophisticated software, but it is generally recognised that all too often it lies unused, or at least under-exploited for want of adequate resources being devoted to the training and support of the software. In the early days of CAAD it proved particularly difficult to overcome the lack of tradition in architecture of capital investment in tools of the trade. Architects were simply not used to buying major...
items of equipment. That an architect may buy a computer system is no longer thought extraordinary, but it often still remains difficult to persuade senior management to provide training and allow staff time away from the need to perform in order to learn to use that system. One architect who invested in a CAAD system even advanced the argument to me that he would not have his staff trained to use the system, because they would then want more money and, even worse, would be recruited by other practices with the system. This may seem to be an absurdly extreme example, and there are of course some architects with very enlightened attitudes towards the training of their staff. However there is undoubtedly a general lack of recognition in the profession of the benefits to be gained by investing in the training and education of staff.

What is slightly more disturbing is that the same basic problems also exist in many schools of architecture. Many schools have struggled to convince their funding authorities of the need to purchase CAD equipment and software. The substantial educational discounts offered by many hardware manufacturers, and even more dramatic discounts available from software suppliers have, to some extent alleviated this problem. However schools without major CAD research units often simply do not have the staff resources to tutor and then support their students in the use of sophisticated software. Educational institutes rarely have the money to send even one member of staff on commercial training courses, and those staff can seldom find the time to learn the intricacies of the software under their own steam. Indeed student users of CAAD often find themselves the most expert in their institute. They may get some basic tuition and simple training but find themselves unable to exploit the software to its best when actually working on their design projects.

Experience of teaching CAAD has generally shown that designers in general and architects in particular can find the process of learning to use computer aided design a rather frustrating experience. Inevitably good CAAD systems are now very sophisticated and thus complex. In recent years we have seen an increasing interest in the whole area of user interface design. This is reflected in studies of the software metaphor (Van Norman 1986), and the packaging of the ergonomics of the screen, keyboard and printing devices into WIMP environments. However, no matter how well designed this user interface, learning to use such systems is non-trivial and deserves careful attention. One of the difficulties here is that CAAD systems must be reactive not prescriptive in their relationship with the user. That is, the system must wait for the user to take the initiative and cannot lead him or her through the processes required to
achieve the desired objectives. Consequently the user must often have a pretty good idea of how the system works as a whole before any real progress can be made.

Often then, the problem facing the new user of a CAAD system is not in terms of understanding the syntax of commands and what they do, but rather knowing how the system works, what concepts it employs and what combination of commands will achieve a particular design end. Such issues become even more critical as we move from Computer Aided Draughting to real Computer Aided Design. The design as opposed to the draughting process is likely to be iterative and the computer based element more integrated into other human activities. Once we are into the realms of intelligent building modelling then the computer tool has so penetrated the architect's fundamental design task that it may become absolutely essential for him or her to have a clear mental model of how the system works and is organised. (Lawson 1983)

However this poses another problem which is often poorly addressed by CAD courses. This concerns the educationalist's model of the software user. All too often CAD training courseware reveals an implicit assumption that there is only one type of user of the system. In practice nothing could be further from the truth. Certainly for example, student users of CAAD are very different in their requirement and patterns of usage to professional architects. We should not of course be surprised at this since the school of architecture studio is not intended to be an accurate model of the professional drawing office. Students are, inter alia, much more concerned with the early phases of design, and less occupied with the business of producing properly co-ordinated and comprehensive sets of production drawings than their professional. Some have drawn the conclusion from this that separate specialised CAAD systems should be developed purely for educational use. This argument seems to us at best not proven for several reasons. Firstly students should not exclusively be focussed on their immediate project but should also become aware of the nature of architecture as it is actually practised, and for this reason should at least be shown and have access to professionally used software. Secondly, the development of complex software is a tremendously labour intensive, highly skilled and costly process. Such an undertaking followed by the consequent problems of marketing, supply and support are almost certainly not viable commercially if directed purely at the architectural education market. Another alternative is the use much more general software of the kind that can be marketed extremely cheaply since it appeals to a very inside audience, but this unfortunately cannot provide specifically architectural
applications. Finally the idea of developing CAAD purely for educational use is based on the assumption that all student users are different from all professional users, and this argument cannot really be sustained after an examination of professional users.

A study of CAAD systems actually in use in professional practice reveals two groups of lessons. Firstly, the range of software varies from two-dimensional draughting through many degrees and variants of three-dimensional capability right up to fully integrated intelligent three-dimensional building design and draughting. The pattern of use of the three-dimensional software in particular also shows enormous variation. Some practices use such software largely as unnecessarily complex two-dimensional draughting, others just for early design visualisation, and others throughout the design process more or less from beginning to end. In many cases the pattern of usage also varies within a practice from job to job. The second set of lessons concern the range of personnel and their roles in the organisation and relationships with the computer system is also very wide. In role terms, users may be design executives, office managers, technicians, draftsmen and even secretaries by original training. In functional terms CAD users can have many different relationships with data. Some may have the full range of involvement, entering data, making design decisions based on evaluative feedback supplied by the system, and plotting drawings. Others may act basically as technicians just entering data from sketches or other instructions supplied to them by designers. Others still may run routines to produce plots, schedules and other forms of output. Senior design staff may choose to look at the data by examining models in perspective or running evaluative routines, possibly experimentally editing data, whilst delegating the bulk of the data input to others.

Some writers on CAAD have expressed their own strong views about the need for systems to be totally accessible to all staff at all times. The evidence from observing practice is that some may be regular, even daily users of CAAD whilst others are occasional or casual users. It is surely not for the system designers to specify how their CAD system should be used, and clearly one single model of a user who is often assumed to be enthusiastic, totally trained, skilled and making daily use all parts of the system, simply will not do. In fact it is possible to see a whole range of subsets of system features on which a user needs training, which will vary according to the motivation and role of the individual user and the organisation of the design office. At Sheffield University we are investigating how it might be possible to allow CAAD users to acquire system skills in a manner more sympathetic to their learning style, more
appropriate to their functional tasks, and more suited to their organisational needs.

Existing approaches to training for CAAD The above very brief analysis shows that many users are unlikely to find the traditional system oriented approach to training particularly satisfactory. In this approach material is prepared and presented in such a way as to follow the structure of the CAAD system. Concepts and commands are introduced in an order and fashion which seems logical to the designers of the system, and usually in some roughly ascending order of difficulty and complexity. Thus training proceeds on a kind of 'walk before you run' basis, with students learning ever more clever tricks lesson by lesson until all the material is covered.

It is difficult for users who only need to use certain features or parts of the system to select from such training courseware, since each lesson often depends on assumptions about the student having followed the course in its predetermined sequence. Designers in general and architects in particular seem to respond particularly badly to this kind of training. This is hardly surprising when we examine the educational style of design courses. Designers are generally educated through a process of 'learning by doing'. They expect to use their creativity to solve a series of problems which in turn bring forward a series of theoretical issues. Thus wading through a mass of theoretical descriptions accompanied by series of predetermined exercises is a most unnatural process. This lesson has had to be learnt in other technical areas of architectural courses such as environmental science and structural engineering. (Lawson 1980) An early set of tutorial exercises for GABLE written at Sheffield University which really failed to recognise this experience have thus proved unpopular and we have learnt several important lessons from this failure.

In the design of these tutorials considerable emphasis was laid on preventing the student from getting lost, and thus each step was described in great detail. Architectural students quite simply found this boring and tedious and two problems commonly arose. Firstly the students reported simply performing the tutorial in rate fashion and failing mentally to absorb the concepts. Secondly, some students rebelled against the rigidity and broke out by developing their own individual variants on the drawings and models they were asked to produce. These students then got lost because the tutorial could no longer describe in sufficiently accurate detail what should happen, or even worse, the students 'design' could not be subjected to the particular tasks demanded later in the tutorial.
Paradoxically it seems that the very students who find CAD most frightening are often also the most likely to rebel against a rigid tutorial.

A new approach to training for CAAD
How then might we be able to provide more suitable and flexible learning methods for students of CAAD? Our own unstructured observations of what students seem to prefer to do, the organised feedback of system users (Lossing et al 1989) and those empirical studies we have conducted which evaluate student satisfaction with training courses (O'Connell 1988) have pointed us in the same direction. All of this data has suggested to us that a 'learning by doing' or 'problem solving' approach to CAD tutorials might be more successful. With this technique students would simply begin to use the CAAD system for their own design purposes and seek instruction and expert help as and when needed. This simultaneously makes the learning immediate and relevant, and also satisfying in that useful and creative design work is being performed by the student. The educational problem with such an idea is that unfortunately it really requires one to one teacher to student tutoring, and consequently demands a large pool of highly skilled experienced users of the CAAD system who are also able to teach. Those of us lucky enough to have fairly large CAAD research units in schools of architecture, as we do in Sheffield, can probably just about provide this from our research students. Clearly however such courses could not economically be mounted in many schools of architecture or in professional practice.

When faced with a shortage of human expertise of this kind it seems sensible to see what computer-based techniques could do to alleviate the problem. Thus we are now engaged on a number of projects to explore intelligent computer-based tutoring and expert guidance systems for teaching CAAD. The idea behind this program is to provide a flexible or open learning system on which the student can draw choosing learning styles and situations at will.

There are a number of key features of the system, known as the ICAD Tutor, which it is necessary to establish before discussing the approach in detail. Firstly it seems sensible that the Tutor should contain knowledge about the CAD system in such a way that this knowledge could be changed to represent a different CAD system, without altering the Tutor software. This attitude towards knowledge implies that the Tutor will be written in a declarative language, whereas CAD systems are almost always written in procedural languages. All this suggests therefore that the Tutor should be separate from, but nevertheless ideally able to interact with, the CAD system. In
our case, for the early stages of prototyping the Tutor, we are writing in PPOLOG on both the IBM PC and Apple Macintosh environments. We are currently developing a tutor for the GABLE 4D Series CAAD system which originated at Sheffield University but is now developed and marketed by GABLE CAD Systems Ltd. In our development environment GABLE runs on an Apollo Domain network which also allows access to the disks by a DOS environment.

The tutorial system will be based around the Concept Network approach, which depends on the idea of expressing items of knowledge and declaring information about their interdependences. This offers three distinct advantages in this context. Firstly, the courseware author can concentrate on high level issues related to specifying knowledge to be taught rather than be caught up in the low level presentation details. Secondly, one investment of authoring time covers a vast range of possible tutorial material which can be generated by the system for each individual's needs. Thirdly, the system can support individualised teaching based on a record of what the student knows as a subset of the overall knowledge involved in the domain.

The most recent approach to the development of intelligent courseware has advocated the representation of the knowledge to be taught as some form of semantic network. (for example the feature networks of Webb, 1988). A tutorial is then drawn from this network as a 'fragment' of knowledge and presented to the student. The student's state of knowledge can be modelled as a subset of this network and can, in principle, be used to guide tutorial selection and assembly and can be updated according to student performance. We therefore propose an intelligent courseware design system consisting of the following modules: (1) a concept network system (2) a tutorial specification system (3) a student record maintenance system (4) a tutorial presentation and performance monitor. A fully working architectural ICAD Tutor would require all four modules to be in place and we have already developed some crude prototypes of each which will be briefly discussed here. However each is a significant project in its own right so we intend to focus primarily on the tutorial specification module in this paper.

A concept network system
The knowledge of the system consists of high level conceptual descriptions of tools, commands, tasks, features of drawings and so on. Distinct in the proposed tutor are three classes of knowledge: concepts of the GABLE tool specifically; more generally about Computer Aided Design;
and most generally about principles of architectural design. These concepts are inter-related as mutually dependent, more or less primitive and so on. Closely connected with each concept is a primitive instructional unit, or topic, from which tutorial sessions may be assembled. Clearly the development of this network and its maintenance is a major project in its own right, and one which is receiving a great deal of interest in the AI community, (see for example, Jacobson and Freiling, 1988).

For the prototype version we expect to develop a concept network covering less than a thousand topics. It seemed essential to be able to include graphical material in these topics and rather than use a word processor with graphical facilities we have chosen to develop the topics actually using the GABLE drafting system. This enables the full range of graphical facilities in the CAD system to be displayed with any tutorial topic. We have written a program to translate GABLE two-dimensional drawing files into a format can be displayed both on the Apple Macintosh and by Turbo-Prolog on the whole range of screens supported by the IBM PC. The Tutorial author will thus rely on his or her own experience of the CAD system about which knowledge is being engineered in order actually to express that knowledge.

Each topic has a title of unlimited length and unconstrained format which is linked to its filename which of course must observe the constraints of the host operating system. After the topics are developed and declared the author must state upon which other topics each topic depends. Some primitive support tools to aid this process have so far been written in Turbo-Prolog. For example a search can be made of all the text in a topic for references to all or part of the names of other topics. Alternatively searches can be made for all the topics which include references to individual words or complete phrases. However the question of the extent to which this process could in reality be automated and just how the concept network is maintained especially as changes take place in the CAD system, although interesting and important issues, are really beyond the scope of this paper.

A Tutorial Specification System
This system must extract a fragment of the concept network which is appropriate to the student’s current needs and state of knowledge. To do this we must assume that the student enquires in some way about one or more topics which exist in the concept network. We have developed a PROLOG algorithm to extract the appropriate fragment using any number of starting points in the concept network. The program proceeds by searching the network for all topics.
upon which the starting topics depend, and then for those topics upon which they depend. The routine proceeds in the classical PROLOG recursive manner until an exhaustive list is obtained with no duplications and so organised that no topic precedes another upon which it depends. We have identified three forms of knowledge needs across all classes of system user, and these give rise to three methods of defining the starting points for the development of tutorials. These may be called: concept based; task based and analogy based.

Concept based
The first appreciate to tutorial generation is driven by a concept based analysis where the student wants to explore a known sub-area of the system's knowledge. For example in GABLE, the student may simply ask about the use of the 'arc' drafting element or the 'align' command. This kind of enquiry is similar in one respect to the on-line manual or help facility, and thus could be triggered by a special keypress then picking or printing to system menus. However, unlike the help facility, the Tutor will prepare and present a complete tutorial including all topics needed to explain the concept about which the enquiry is made. This would thus offer a learning style in which the student simply selected at will from system menus and gradually learnt about the system, from the resultant tutorials.

Task based
More usually however the novice user is engaged in a specific task which may require a range of concepts which may not necessarily be known to the user. In the task-oriented approach the student would seek instruction on performing a particular task such setting up a perspective views or plotting a drawing file. Some high level kind of interface would be required enabling the CAAD user to ask a question of the Tutor such as "how do I get a perspective view of this design?". From this inquiry we would require the ICAD Tutor to recognise that the user may need to know about a number of commands which, for example, set the view point, target point, lens angle and drawing characteristics.

Design analogy based
Finally, and most problematically, the user may not even knows exactly what tasks must be performed in system terms to produce an effect in design terms. Providing a tutorials to meet this need could be achieved by building a library of designs using the CAD system, and in some way describing to the Tutorial system how these designs were created. A provision could then be made to allow the user to indicate that a desired design is analogous in some way to another design about which the system already knows. This idea poses some interesting problems and at first sight seems
extremely ambitious. However we have already developed a prototypical solution to the problem of building the library of designs together with knowledge about the way they were created. Virtually all good CAD systems provide some means of recording the actions of users although the precise mechanism varies from system to system. Of course such a technique is widespread and is even included in the word processor on which this paper is written. GABLE is no exception and includes a facility known as 'record/playback' which streams all the user's inputs and responses into a file, which can later be used a kind of macro for repeating these actions optionally in a parametric manner. We have provided a method of reading this data into the Tutorial builder to provide a list of all concepts around which a tutorial will be required before the design in question could be created.

In all three cases above the first difficulty for the Tutorial system is to recognise the user's needs. In part the student model is helpful as the system can, infer the student's current state of knowledge, however rather more elaborate procedures are required to extract the appropriate fragment of the total system knowledge applicable to a task based or design analogy based, and in particular these require a more carefully considered approach to the presentation of the tutorial. Thus to complete the ICAD tutor we must consider the final two components; the student model and the presentation module.

A student record maintenance system The student is modelled as a simple subset of the system concept network. This model has great predictive value towards monitoring the teaching process, and must allow for the great range of student types identified in our earlier analysis. The development of more sophisticated user modelling facilities is central to all intelligent human computer research, but is particularly pertinent to the tutoring domain. (for a recent critique see for example Laurillard 1988). However clearly the student record would at least contain information which could reveal those topics the student has reviewed and how often and which order. For our purposes analysis of the student records may therefore also yield information indicating those aspects of the CAD system which for example are found difficult, or remain unused and so on.

A tutorial presentation and performance monitor
This module will accept enquiries from the student, present the tutorial, assess student progress and update the student model. The presentation and performance module could itself usefully be intelligent in its control of user interaction, and indeed this is the level at which most tutoring effort
has so far been expended. We have already provided two methods by which the student may break out of a prepared tutorial, and seek information in a more open ended fashion. Firstly it is possible to browse the whale concept network from any starting point and not only to view a topic but to call for menus of those topics which directly explain the current topic, or alternatively which the current topic helps to explain. In addition we have added a Hypertext capability to the system in which any card or phrase in the current topic can be highlighted by pointing with the mouse and this used as a new starting point in the browse mode. Thus a student can follow a tutorial but optionally gather other information under his or her control at will.

For the prototype we propose only a very simple solution to the problem of judging the student with pre-packaged critical tests. However the work already described allowing the ICAD Tutor to examine protocols may well allow for online critiques of student performance by the tutorial system, and this will form the basis of some future work.

These ideas have spun off a number of other research directions upon which work has now begun. The library of designs with their associated protocols can also be used as a pool of examples which could be called down to illustrate a particular topic in either a concept or task based tutorial. Thus the tutor could search all the protocol files for instances of the topic and select a number of examples as illustrations.

More ambitiously, we have begun examining the possibility of using an expert system to examine the student protocol in order to offer help when things go wrong. Inexperienced students undergoing the task and design based tuition advocated here are quite likely to get into difficulties by trying out system features beyond their current understanding. Typically students may present problems to tutors of the farm: "Why is the screen blank?" Such problems are, in principle, quite amenable to conventional expert system analysis. In GABLE the expert would need to know if the student was in plan, elevation or perspective view, and then in each case could suggest one or more possible reasons for the blank screen, and then offer instruction on how to remedy the situation.

Conclusion
We firmly believe we have devised an open learning environment for CAAD which is likely to be both more appealing to the student and more cost effective than traditional training techniques. Preliminary investigations into several leading CAAD systems have shown them all to be capable of linking satisfactorily to our ICAD Tutor, and
Early tests with users are encouraging. The programme has now received some initial funding and we hope to be able to develop and test these ideas more thoroughly.

References


Laurillard, D. 1988 The pedagogical limitations of generative user models. Instructional Science 17, 235-250


Lawson, B. R. 1983 How Designers Think Architectural Press


O’Connell, K. 1988 CAD: A design revolution? BA Thesis Department of Psychology University of Sheffield


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

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