This paper discusses recent developments in Intelligent Tutoring Systems (ITS) which emphasise a constructivist approach, giving control of the learning to the learner. These are better described as Adaptive Learning Environments (ALEs) and include help systems, coaches, discovery environments, microworlds and other variants. A number of these systems are discussed in terms of their teaching strategies. The particular applicability of ITS techniques to learning procedural knowledge in problem solving, student centred and investigative domains places the development of ALEs on a sound footing of constructivist theory and assures continued research in classroom applications.

A STORY ABOUT TEACHING AND LEARNING IN THE MATHEMATICS CLASSROOM

Some years ago I taught in a secondary school where the Deputy Principal was very keen on what he termed 'groupwork'. He had a vision of classes being conducted throughout the school where the students worked collaboratively, sharing their ideas with each other, experimenting, learning procedural skills and working cooperatively with the teacher in the role of a facilitator. His first step in implementing this very sound idea was to issue a directive that all classrooms were to have the tables arranged in small islands or clusters so that a number of students could work together. Now the mathematics staff, to which I belonged, were not pleased at all. Groupwork they claimed was rarely appropriate to their subject. Better to have the students sitting in pairs, in columns facing the front so that there was a minimum of chatter in the room. They believed that naturally the disruptive students gather together in a group situation, and little work is done unless they are separated. Mathematics they said was a subject that was best taught in a didactic, teacher-centred way for most of the time.

The problem of course is that for cooperative learning to work properly, the teaching style and student activity must be as appropriate as the seating arrangement. It seems that many teachers, perhaps particularly mathematics teachers, exclusively use a teacher-centred approach, and think of inquiry learning as inapplicable to their subject, and as having the potential to be an unproductive activity likely to cause discipline problems.

TWO APPROACHES TO TEACHING AND LEARNING

There are two approaches to teaching and learning exemplified in the discussion above. A behaviourist approach adopted by the Mathematics teachers on one hand, with their structured learning strategy and emphasis on routine, sees learning as an organised transfer of knowledge. The exploratory view, on the other hand, is supported by collaborative learning and groupwork, and suggests the now widely accepted constructivist approach where learning is regarded as the formation of mental models or 'constructs' of understanding by the learner. In this view of learning, the students actively build knowledge based on previous understanding by dynamically interacting with the learning media. The teacher-mediated construction of problem solving skills and procedural knowledge is a relatively new direction for mathematics education and has major implications for curriculum and assessment. Its implementation through teaching strategies which support this view will undoubtedly take some time to filter through to be commonplace in all classrooms. (see Hand, Lovejoy & Balaam, 1991)
My approach then and now is that there are many opportunities to use inquiry learning in the Mathematics classroom, and that it can be made to work very effectively. I have seen a number of very successful classroom applications of the technique. Undoubtedly, for much of Mathematics teaching the traditional form of classroom interaction which places the teacher as a giver of ‘external’ knowledge (Cobb, Wood, Yackel & McNeel, 1992) is a simple and effective teaching style which provides a systematic way of covering a demanding syllabus. While assessment procedures favour ‘content’ over ‘process’, teachers will tend to adopt a teaching style that achieves the best assessment outcomes. Besides these considerations, there is little doubt that the behaviourist learning paradigm is particularly useful for revising previous work, practising graded examples and diagnosing student misconceptions. Once students have constructed their understanding of the material through investigative sessions, a structured tutorial is a useful way of reinforcing understanding through practice. Also, less able students find difficulty in forming their own understanding of concepts and need more teacher-imposed direction. Such students respond well to, and expect their teachers to provide, classroom discipline and well-structured lessons. Importantly, classroom experience tells us that neither approach is the better one all of the time, and a variety of teaching methods can be used depending on the content, the students and syllabus constraints.

These two perspectives of teaching and learning are similarly represented in the field of Computer Assisted Instruction (CAI), with notable examples being microworlds such as Logo as an implementation of the constructivist ideology and drill and practice programs as a proponent of the behaviourist approach. In this article I wish to explore the area of Intelligent Computer Assisted Instruction (ICAI) as it relates to these models of learning, and in particular describe a number of variants of intelligent tutors which exemplify less directive and prescriptive approaches to ICAI.

INTELLIGENT TUTORING SYSTEMS AND ADAPTIVE LEARNING ENVIRONMENTS (ALES)

An Intelligent Tutoring System (ITS) is computer based learning system designed for the teaching of content from a syllabus and incorporating program-
ultimately satisfy goals, and the user-model and expert-model are defined in terms of satisfied subgoals. Genie’s responses depend upon the question intent, the computational goal, the context and the user model. The responses are formulated by three components: an understander which phrases English questions, a plan analyst which locates the best plan for the goal and interfaces with the user model, and an explainer which introduces the related plans as enrichment.

Intelligent Help Systems (IHSs) (see Breuker, Winkels & Sandberg, 1987), have been devised to support the task performance of users of interactive computer programs. These are distinguished from an ITS in that the teaching from IHS basically occurs only occasionally when the user requires assistance. An IHS can be called upon to help or may interrupt when the user is making frequent or conceptual errors. This is a useful feature as novice users often do not know when they are going wrong or how to describe the problems they encounter. Essentially an IHS is a coach, taking a more passive role than a tutor. A notable example is the Eurohelp project, a help system for Unix-Mail (Winkels & Breuker, 1990).

Through the development of a Learning Companion System (LCS), Chan and Baskin (1990) explore the notion of a learning companion or collaborative partner. An LCS involves three parts: the human student, the computer learning companion, and the computer teacher. In an LCS there is, apart from the computer teacher, a separate part of the program which acts as a companion learner, much as a peer would, which ‘learns’ along with the student. The companion motivates the student, stimulating learning through collaboration and competition. The domain is indefinite integration, dealing with a variety of techniques including the method of integration by parts. The authors suggest three stages of learning:

- Firstly, they observe an imitation stage where the student will mimic the teachers example;
- Secondly a developmental stage where the fundamentals of the technique(s) are understood; and
- Lastly, the student enters an integration stage where the student has mastered the notation and attempts to apply the principles of the techniques to new situations.

Following these three learning steps the LCS is designed to firstly present coursework in a traditional question-answer format, then allow the student to solve problems with the LCS where ongoing hints and prompts are given, and finally introduce a wider range of more difficult questions which rely on a number of more complex techniques as well as the student’s ability to choose the appropriate method of solution to the problem.

Working collaboratively is a practice that many teachers have used in their classrooms to good effect. To apply CAI to this idea, Chan and Baskin suggest a three tiered approach to intelligent tutoring with the computer as the student’s learning companion and the teacher as the overseer of the learning situation. The effectiveness of the learning companion relies on the companion having a knowledge acquisition rate closely matched to that of the learner, so that the companion uses the same heuristics in the problem solving process. To learn a heuristic the companion requires a notation powerful enough to describe the generalisation. Clearly such systems are only possible in well-ordered domains, and even that of indefinite integration appears somewhat complex. Chan and Baskin (1990) note that the level of sophistication required is beyond current learning systems. Undoubtedly the computer would not be as effective a partner in learning as another student, so it is a notion whose value is not immediately apparent.

**Domine** (DOMain INdependent Instructional Environment) is a tutoring system which uses multiple teaching and assessment strategies in a number of different domains. ‘Complete domain independence is not feasible...’ (Spensley et al., 1990, p.189), so **Domine** has been created to teach people to carry out tasks in three specific areas: telephone exchange maintenance, office automation and system administration. The **Domine** system was designed to teach computer-based procedural skills. In constructing the system the authors make the distinction between three types of knowledge: Conceptual, procedural and interface knowledge. The student model consists of a number of numeric tags associated with the components in the system, and represent the system’s belief about the student’s knowledge.

Additionally, it contains a ‘dialogue history’, a record of the computer-student interaction, which provides a basis for the teaching strategies and assessment decisions. The system has eight teaching and assessment strategies, as well as a facility for the student to revise or fast-forward the lessons. The strategies are:

- **Cognitive apprenticeship**, where the student firstly learns by watching the expert, then performing selected parts of the task under directorship of the expert, and lastly perform an entire problem-solving task. Successive refinement, where a number of levels of detail are employed in explaining a topic depending on student understanding.
- **Discovery learning**, where a particular task is selected for the student which allows the learner to develop from their current knowledge state.
- **Discovery assessment**, which follows the discovery learning strategy and monitors the students progress with this strategy.
- **Abstraction**, which provides the learner with an overview of material rather than concentrating on specific tasks.
- **Socratic diagnosis**, where the tutor presents material which enables the learner to identify flaws in his/her own understanding of the knowledge domain. (This technique is much used in ITS, and **Domine** detects errors and infers a likely path to the error from the student model and presents material on this basis). Practice, where students familiarise themselves with new material through repetition.
- **Direct assessment**; and finally **Strategy assessment**, where **Domine** decides upon the use of one of these strategies based on a balance between teaching and assessment, a student’s success with a particular strategy, and a student’s own preference.

The authors claim that **Domine** does not contain a model of the teaching/learning process but uses a broad tutorial style of ‘decreasing intervention’, and continually assesses the effectiveness of the teaching strategy on this basis. However, it could be argued that the effective use of multiple teaching and assessment
strategies is in itself a model of teaching and learning. The teaching strategies are described by the authors as 'course-grained' due to the fact that the system is partially domain independent. Dominie was implemented in LPA Prolog on an IBM PC.

Dugale (1992) has illustrated by a number of examples of educational software, including a fractions tutor, Green Gloves, and another hypothetical fractions tutor, that perhaps the most important consideration in the design of educational software is that the instruction is based upon a sound cognitive model. Thus the knowledge domain needs to be presented to the student using influential principles which reflect the cognitive processes the student uses in the acquisition of knowledge. Dugale believes that this design principle overrides other considerations, such as instructional style in the software or standardisation of the student-computer interface. The hypothetical fractions tutor features direct coaching to improve student performance on solving problems and the traditional focus on memory is replaced by one of reconstruction skills. He suggests a number of instructional design principles: design instruction around the subject matter (domain independence is not feasible or desirable); allow for creativity in the way the material is presented and flexibility/insight in the manner in which the student may solve the problems; and look beyond immediate objectives and student mastery to the attitudes, perspectives and previous experience of the individual student.

The suggestions for instructional design proposed by Dugale (1990) are reminiscent of those for good CAI, or good advice for beginning teachers. For ITSs to ultimately be used effectively in the classroom, a range of practical considerations must take priority over research goals. Incorporating advanced features such as authoring may make implementation in non-trivial domains on microcomputers found in schools impossible. The use of multiple teaching strategies may complicate the teaching and confuse the student, who might be better served by an emphasis on a simple and intuitive interface which minimises 'cognitive noise'. For ITSs to be commonly used in classrooms, the overall aim must be to create a simple to use, effective tutor which supplements the teacher's work. Designers of such systems will need to concentrate on issues related to their actual use.

Some researchers in the field (Thompson, 1990) are recognising that demands on teachers' time and energy are such that their criteria for selecting software for use in their classrooms will emphasise practical concerns above all else. Considerations such as ease of use, appropriateness to the syllabus and the program's ability to keep students busy are more likely to attract the teacher to the software than a package which promises to challenge traditional teaching strategies and classroom methodologies. For many existing ITSs to be successful in the classroom, the learning styles adopted by the classroom teacher will need to alter to accommodate the new technology. McKendree, Ranilnisci and Atwood (1992) see the problem as one of technology transfer, where the new technologies provide situated learning opportunities or cognitive apprenticeships and call for new methods on the part of the teacher in charge.

THE ROLE OF THE MODELLER
Student modelling in an ITS facilitates the tutoring flexibility as well as individualising the tutorial discourse. The implementation of student knowledge tracing to provide appropriate questioning/tutoring as well as remediation/extension creates this flexibility. Diagnosis of student errors by the modeller at varying granularity (or level of detail) enables the tutor to seize opportunities to remedy student misconceptions through remedial tutoring, or to alter the instructional strategy of the lesson. Student knowledge tracing is therefore central to the use of multiple instructional techniques in an ITS.

The role of the student model in an ALE is essentially to keep a record of the student's understanding as the lesson progresses, and it does so on the basis of student responses. Exactly what information needs to be kept, how it is recorded and how it affects the way the computer interacts with the student is a matter of considerable academic attention, and forms a major part of the study of ITS in its own right. Essentially, the form and function of the modeller is dependant on the adaptivity of the environment as well as the model of learning upon which the system is based.

There is some debate about the need to model the student's knowledge at all, because as we do so, we are implying that there is some expert way of solving a problem, and we are guiding the student on a path which may stifle creativity and experimentation. Some ITS implementations which place the responsibility for learning with the student are actually unintelligent tutors (Kintsch, 1991) in that the modeller is dropped from the system architecture, the notion being that the environment should create learning situations and provide the assistance and the tools needed to solve problems rather than structure learning through error diagnosis or knowledge tracing. Diagnostic feedback often replaces the use of a student model (see Watanabe et al., 1991).

CREATING AN INTELLIGENT MICROWORLD
Suppose we wished to add some intelligence to the Logo microworld. We might decide to divide the screen into two distinct areas, the usual Logo interface for students to write instructions and obtain the appropriate response from the language, and a prompt area. The Logo area behaves in exactly the same way as Logo usually does. We aim to improve the environment in two ways: Firstly it recognises incorrect syntax and offers suggestions. When a student makes a detail error, for example missing a bracket, leaving a space out or typing something like 'RIGHT TURN 90' instead of 'RIGHT 90', it suggests the correct syntax in the prompt area. The student is then able to act on the suggestion of the computer, or continue at will. Now with this first alteration we have not introduced any intelligence - the computer still remains ignorant of the student user. We have not restricted the freedom of the student in the environment at all, but we have introduced some low-level assistance in allowing the student to concentrate on planning rather than syntax.

The second improvement to the Logo environment is the addition of a student model, so that as the student uses Logo, a record is being made of the student's use of keywords and sequences of keywords. The purpose of this is to provide an appropriate level of prompting when a mistake is made, or even to suggest alternatives to sequences of keywords. For instance, if the modeller records commands being used in an inefficient way, the computer might 'suggest' alternatives in the prompt area. When keywords are used incorrectly, remediation follows, but when a detail error is made using a keyword that the modeller believes has previously been mastered by the student, clearly no tutorial is needed. The basis for this is the belief that the level of remediation indicated for a mistake is as dependant on the student-history as it is on the nature of the error. A modeller which
constructs this history throughout the lesson is thus able to provide information for the tutor to provide individualised instruction.

The role of the modeler is minimal in this hypothetical implementation. We could extend the environment to include a facility to generate exercises, and use the data held in the modeler to dictate the difficulty and type of question posed by the Logo tutor. However, this would completely change the nature of Logo as a non-directive environment. Perhaps it is better that the teacher takes the role of providing graded exercises and tasks for the student in teaching the Logo language, and allow the computer to provide the assistance at a finer level of detail through prompting and remediation, in the manner of Chan and Baskin (1990) suggested earlier in this article. After all, a human teacher is likely to have a greater understanding of student personalities, learning styles, classroom management techniques and classroom dynamics than the computer tutor for a long time to come.

CONCLUSION

Intelligent courseware is distinguished from other courseware in that it creates a dynamic model of the student's understanding and compares this with a ideal model, an expert solution to the problem. Thus the distinction between unintelligent and intelligent software is that in the latter the system is building a model of student knowledge as the lesson progresses, and in the case of tutoring, uses this model to generate exercises and as a basis for the selection of new material. Such systems are often called adaptive as they are capable of adjusting their interaction based on a changing record of student knowledge.

Artificial intelligence techniques may be applied to a range of human-computer interactions from the most adaptive or system controlled (e.g. Anderson’s Geometry Tutor. Anderson, Boyle & Reiser, 1985) to the most reactive or student controlled of a programming environment. This is a clear indication that research in ALEs will provide a range of computer aided learning solutions, based on learning theories, and performing specific functions as solutions to practical problems. The particular applicability of ITS techniques to learning procedural knowledge in problem solving, student centred and investigative domains places the development of these ALEs on a sound footing of constructivist theory and ensures continued research in classroom applications.

The spectacular lack of success of tutorial software in schools has been well documented, as Sinclair (1993) writes '[...they] often had glaring weaknesses in instructional design, were not completely relevant to the class curriculum and tended "to be little more than expensive page turners".' (Sinclair, 1993, p 22). ITS promise to deliver far more than this, and with the availability of powerful computers in schools, are looking to be a distinct possibility for the near future.

REFERENCES