Software evaluation as a focus for teacher education

David Squires
School of Education
King's College, Waterloo Road, London SE1 8WA, UK

Abstract
In this paper the use of educational software evaluation as an organising theme for IT related teacher education is considered. Effective educational software evaluation involves a high degree of knowledge and understanding of teaching, learning and curriculum design. Developing software evaluation skills can provide rich opportunities for teachers to acquire knowledge and understanding in these areas. The relationship between software evaluation and teacher education is discussed in terms of a novel software evaluation paradigm, the 'Perspectives Interactions Paradigm', which adopts a situated view of teaching and learning. This discussion provides a basis for proposing a strategy for a software evaluation based school-focused approach to teacher education in the use of IT.

Keywords
Secondary education, professional development, teacher education, evaluation/summative, research, software

1 INTRODUCTION

Teachers in all disciplines are now commonly expected to use information technology (IT) to support teaching and learning. For example, in delivering the National Curriculum for England and Wales, teachers are expected to provide pupils with 'opportunities, where appropriate, to develop and apply their IT capability in their study of National Curriculum subjects' (Department for Education and Welsh Office, 1995, p. 1). To do this effectively teachers need to develop skills and awareness relevant to the use of IT in their disciplines. Effective educational software evaluation involves a high degree of knowledge and understanding of IT related teaching, learning and curriculum issues. Developing software evaluation skills can provide rich opportunities for teachers to acquire this knowledge and understanding and in this paper software evaluation as a focus for teacher education in the educational use of IT is discussed.

2 APPROACHES TO TEACHER EDUCATION

The course-based approach is a common model for in-service education. Morant (1981) describes it as a 'top-up' approach. It assumes teachers can be educated by correcting deficits in their knowledge. Most courses are short term with no plan for individual development and Millar (1990) points out that this approach is compatible with the deficit model of teacher education (Eraut, 1987). This model assumes that teachers are deficient in skills due to limited initial training, or lack knowledge because they have failed to keep up to date.

School-based teacher education programmes can be described as being initiated, planned, and executed by school staff for teachers actually serving in the school (Millar, 1990). The rationale of this approach is that it is easier to identify relevant needs (Henderson, 1979; Morant, 1981). This approach links to two further paradigms for teacher education identified by Eraut (1987); the change paradigm and the problem solving paradigm. The change paradigm requires teachers to accommodate to local and nationally imposed changes, for example, the introduction of a statutory national curriculum in England and Wales. The problem solving paradigm is geared towards dealing with issues and developments in teachers' schools.

Millar (1990) cites Howey (1986) as describing the priority of a school focused agenda as 'the improvement of those conditions and processes which most directly affect the quality of education of students within a given school' (p. 24). This approach typically involves contributions from staff within the school and external consultants, and course attendance. School-focused education attempts to realise the advantages of the school-based approach and minimise the disadvantages of courses. This approach clearly relates to the problem solving paradigm. It also relates to a fourth paradigm identified by Eraut (1990); the growth paradigm in which teacher education is seen in terms of personal development and career needs.

3 EDUCATIONAL SOFTWARE EVALUATION

It is now commonly advocated that cognition and learning are situated in specific learning contexts (for example, Brown, 1989; Pea, 1993; Clancey, 1994). Brown refers to learning being 'idexicalised' to the context in which learning occurs, meaning that learning is inherently associated with the context in which it takes place. This implies that it is not possible to evaluate an educational artefact, such as a software package, as an educational device in its own right; it is only possible to evaluate the actual or perceived use of a software package in an intended or existing educational setting. Thus a situated approach to software evaluation should be adopted.

3.1 Predictive and interpretative evaluation

Attention to context is straightforward when the actual use of software is being evaluated as the evaluation is an interpretation of observed use. Such evaluation can be called 'interpretative evaluation'. In contrast, 'predictive evaluation' is concerned with the prior evaluation of a software package before it is used with students. The use of a package is imagined in intended learning situations, i.e. the evaluator mentally simulates the use of the package so that an attempt can be made to evaluate the package in context.

3.2 Common approaches to educational software evaluation

Checklists and frameworks are the conventional software evaluation tools, but, as discussed below, they do not allow a situated perspective to be adopted.

Checklists for software evaluation

In a critical examination of the checklist approach to predictive evaluation McDougall and Squires (1995a) identify a number of problems. These problems are symptomatic of a failure to adopt a situated perspective. Some stem from a focus on the software application as an object of evaluation in its own right rather than the evaluation of its use in context. Others indicate that the diversity and complexity of the classroom, and the teacher's role in managing this, do not feature in the design of checklists. Also, evaluation in different subject areas requires different criteria to be used (Komoski, 1987).

Frameworks for software evaluation

Many writers have advocated the use of frameworks to describe the use of educational software. For example, classification by category (Pelgrum, 1991), description of the roles that it is intended to fulfil (Taylor, 1980) and links to commonly accepted educational rationales (Kemmis, 1977).

The focus in classifying by application type is on the software package itself, without consideration of its use in particular educational settings. Classification by educational role emphasises the way software is intended to perform rather than the roles of the teacher and the learner. The use of educational software is conceived primarily from the designer's perspective, which does not encourage a situated approach to evaluation with the designer's decisions interpreted in the context of the use of a package. Links to educational rationales encourages a view of software as belonging exclusively to one paradigm, hindering its evaluation in a variety of contexts. Classification by software environment promotes a superficial distinction between software in terms of attributes rather than context.

3.3 The 'Perspectives Interactions' paradigm

Squires and McDougall (1994) have proposed a situated paradigm for evaluating educational software which is based on the mutual 'interactions' between the principal 'actors' involved in the classroom use of software. Three actors contribute to the learning situation of a package; two live actors (teacher and student) and one passive actor (designer). The situation in which the package is used is defined by the interactions between these perspectives.

The student-teacher perspectives interaction

The interaction between the perspectives of the student and the teacher explicitly relates to the situation in which learning takes place. It implies a consideration of how the introduction of computers into the classroom may change the distribution between the teacher and student(s) of responsibilities for teaching and learning.

There are many reports of software use acting as a catalyst for a wide variety of off-computer activities. In these situations teachers assume the role of manager and resource provider. Collaboration between peers, particularly in small groups, is very important when IT is used (Hoyles, Healy and Pozzi, 1994), notably in terms of enhanced peer discussion (Chatterton, 1985). When IT is used in the classroom teachers can be regarded as managers and supporters, as opposed to directors, of student-focused activities.

The student-designer perspectives interaction

The interaction between the student's and designer's perspectives relates to how the use of the software can aid learning. A tenet of a situated view of learning is that it takes place in a constructivist fashion. An essential aspect of an evaluation of the use of a software package is the extent to which it can support a constructivist approach.

Honebein, Duffy and Fishman (1993) have argued that authenticity and stimulus or concept complexity are crucial in constructivist learning environments. They identify three conditions for authenticity: a sense of ownership of learning, engagement in global activities which go beyond mastery of local skills, and use of multiple perspectives in problem solving.

The teacher-designer perspectives interaction

The teacher-designer perspectives interaction relates to the designer's notion of the curriculum relevance of the intended use of the package. Teachers need to appreciate this and assess how valid it is in personal terms. Evaluation with respect to this interaction is a question of how the designer's and evaluator's perceptions of the curriculum match.

Curriculum issues may be explicit, implicit, or even absent, in the design of educational software. Explicit curriculum issues are evident in packages that have been designed for a defined course or curriculum. Implicit curriculum issues stem from cultural assumptions made by designers, for example, the American bias of many CD-ROMs. Absence of curriculum issues arises when software not originally intended for use in education is used in schools, for example, wordprocessors. Assessing software with explicit curriculum aims simply requires a comparison of these aims with a syllabus. Evaluating software which initially has no curriculum aims requires some imagination as to whether and how the software might be used in an educational context. In this case it is not a question of mapping the teacher's intentions onto the designer's; rather it is a matter of assessing how teachers can impose their perceptions on the use of software.

4 SCHOOL-FOCUSED 'PERSPECTIVES INTERACTIONS'

The 'Perspectives Interactions' paradigm has been used successfully as part of a teacher education course (McDougall, 1995b). The situated nature of the paradigm also makes its use with a school-focused approach appropriate. Both predictive and interpretative software evaluation can feature in this approach. Predictive evaluation brings a realism to courses by relating software to classroom use. Interpretative evaluation can assist reflection on practice.

4.1 The student-teacher 'perspectives interaction'

The student-teacher perspectives interaction relates to classroom practice. As such there is a natural link to interpretative evaluation of the use of software in classroom settings. School-based work which involves teachers evaluating practice in their own schools will provide the most authentic environment for them to appraise their own practice or work by colleagues.

4.2 The student-designer 'perspectives interaction'

The student-designer perspectives interaction implies that teachers should be able to appreciate the significance of theories of learning in software design. Courses provide the best forum for presenting a wide range of learning theories. Teachers can attend presentations by experts and use a broad range of software that they would not typically have access to in their schools. Work located in schools could augment course-based work. For example, evaluation groups, perhaps chaired by an 'expert' from the course, could meet to appraise the learning theories implicit in software used in the school. There is scope for linked interpretative evaluation, using the idea of observing the use of software as a 'window on the mind' (Weir, 1985).

4.3 The teacher-designer 'perspectives interaction'

In the teacher-designer perspectives interaction the relative emphasis on school-based and course-based approaches will depend on the extent to which the school follows a prescribed curriculum. When there is a national curriculum, as in England and Wales, there may be more emphasis on a course-based approach. However, there is always a need for some school-based work. For example, the appropriate environment for developing evaluation skills for the assessment of implicit curriculum issues is the school itself in ethnically mixed schools.

5 SOFTWARE EVALUATION FOCUSED TEACHER EDUCATION

The analysis in Section 4 indicates that considering the 'Perspectives Interactions' paradigm gives a comprehensive framework for coherent school-focused programmes consisting of:

- School-based interpretative evaluation of classroom practice of the use of software.
- Courses on learning theories, perhaps linked to school-based evaluations.
- A mix of school and course-based work to develop curriculum evaluation skills.

A programme incorporating all of these elements would provide coverage of the three main issues in teaching practice - pedagogy, curriculum and learning in authentic contexts.

6 REFERENCES

Brown, J.S., Collins, A. and Duguid, P. (1989) Situated Cognition and the Culture of Learning. *Educational Researcher*, **18**, 32-42.

Chatterton, J. L. (1985) Evaluating CAL in the Classroom, in I. Reid and J. Rushton (eds.) *Teachers, Computers and the Classroom*. Manchester University Press, Manchester.

Clancey, W.J. (1994) Situated cognition: how representations are created and given meaning, in R. Lewis and P. Mendelsohn (eds.) *Lessons From Learning*. North-Holland, Amsterdam.

Department for Education and The Welsh Office (1995) *Information Technology in the National Curriculum*. HMSO Publications Centre, London.

Eraut, M. (1987) In-service Teacher Education, in M. Dunkin (ed.) *The International Encyclopaedia of Teaching and Teacher Education*. Oxford University Press, Oxford.

Henderson, E. (1979) The concept of school-focused in-service education and training. *British Journal of Teacher Education*, **5**, 1, 17-25.

Honebein, P.C., Duffy, T.M. and Fishman, B.J. (1993) Constructivism and the Design of Authentic Learning Environments: Context and Authentic Activities for Learning, in T.M. Duffy, J. Lowyck and D.H. Jonassen (eds.) *Designing Environments for Constructive Learning*. Springer-Verlag, Berlin.

Hoyles, C., Healy, L. and Pozzi, S.(1994) Groupwork with computers: an overview of findings. *Journal of Computer Assisted Learning*, **10**, 4, 202-215.

Howey, K. (1986) School Focused In-service: Synthesis Report, in D. Hopkins (ed.) *In-service Training and Curriculum Development: An International Survey*. Cambridge Croome Helm: London.

Komoski, P.K. (1987) Educational Microcomputer Software Evaluation, in J. Moonen and T. Plomp (eds.) *Eurit86: Developments in Educational Software and Courseware*. Pergamon Press, Oxford.

McDougall, A. and Squires, D. (1995a) A critical examination of the checklist approach in software selection. *Journal of Educational Computing Research*, **12**, 3, 263-274.

McDougall, A. and Squires, D. (1995b) An empirical study of a new paradigm for choosing educational software. *Computers and Education*, **25**, 3, 93-103.

Millar, L.C. (1990) *In-service Education and Information Technology*. MA dissertation, King's College London.

Morant, R. (1981) *In-service education within the school*. George Allen and Unwin, London.

Pea, R. (1993) Practices of Distributed Intelligence and Designs for Education, in G. Salomon (ed.) *Distributed Cognitions: Psychological and Educational Considerations*. Cambridge University Press, Cambridge.

Pelgrum, J. and Plomp, T. (1991) *The Use of Computers Worldwide*. Pergamon, Oxford.

Squires, D. and McDougall, A. (1994) *Choosing and Using Educational Software: a Teachers' Guide*. Falmer Press, London.

Taylor, R. P. (ed.) (1980) *The Computer in the School: Tutor, Tool, Tutee*. Teachers College Press, New York.

Weir, S. (1985) *Cultivating Minds: A Logo Casebook*. Harper and Row, New York.

7 BIOGRAPHY

David Squires is a Senior Lecturer in Educational Computing in the School of Education, King's College London. He has extensive experience of design, development and use of software in schools and universities. He is a past co-director of the Computers in the Curriculum Project, a UK national curriculum development project featuring the use of educational software in the sciences and humanities. He currently directs a British Library research project about use of IT-assisted information systems in academic research.