ELSEVIER

Interacting with Computers 11 (1999) 467–483

Interacting with Computers

Predicting quality in educational software: Evaluating for learning, usability and the synergy between them

David Squires[a,*], Jenny Preece[b,1]

[a]School of Education, King's College, Waterloo Road, London SE1 8WA, UK
[b]Department of Information Systems, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250, USA

Abstract

Teachers need to be able to evaluate predictively educational software so that they can make decisions about what software to purchase and how to use software in classrooms. The conventional approach to predictive evaluation is to use a checklist. We argue that checklists are seriously flawed in principle because they do not encompass a consideration of learning issues. More particularly they fail to adopt a socio-constructivist view of learning. We propose an approach that adapts the idea of usability heuristics by taking account of a socio-constructivist learning perspective. This leads to a set of 'learning with software' heuristics. A notable feature of these heuristics is that they attend to the integration of usability and learning issues. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Educational software evaluation; Usability; Constructivism; Checklists; Heuristics

1. Introduction

Using educational software requires teachers to decide which software to use with their students, for what purpose and in what situations, i.e. to conduct predictive evaluations of the use of educational software. Informal predictive evaluations rely on past personal experience to make value judgements about the quality and potential use of an educational software application before using it in the classroom. However, it is difficult to predict classroom interaction, especially when much of the best educational software is designed to support teaching and learning in new and innovative ways, such as by developing and exploring models (e.g., [53]) or by supporting collaboration at a distance via the internet [46]. This problem is even more severe for inexperienced teachers. A well developed

* Corresponding author.
E-mail addresses: david.squires@kcl.ac.uk (D. Squires); preece@umbc.edu (J. Preece)

PII: S0953-5438(98)00063-0

predictive evaluation method, which can be used to review software in advance of taking it into the classroom, will help to alleviate this problem and provide teachers with a good basis for making initial decisions about which software to use with their pupils.

The aim of this paper is to suggest an approach that provides predictive evaluation guidelines for teachers, which systematically capitalise on past experience while taking cognisance of a socio-constructivist view of learning. Formal predictive evaluations are typically based on the use of a checklist, but we argue that these fail to take account of the widely accepted view of learning as a socio-constructivist activity. Our approach is to adapt the notion of 'heuristic evaluation' introduced by Molich and Nielsen [33] as part of a usability evaluation exercise. Heuristic evaluation is done by experts (in this case, expert teachers) using a set of guidelines, known as 'heuristics'. The purpose of the heuristics is to encourage evaluators to focus systematically on all the important aspects of the educational software design. Using the heuristics prevents evaluators inadvertently forgetting to cover part of the evaluation. The heuristic evaluation process requires teachers to review the software, and from their knowledge of how they would present the software to pupils and how pupils learn, the teachers judge the suitability of the software for its intended educational purpose.

We provide a synopsis of the socio-constructivist view of learning to identify salient learning issues that should feature in educational software evaluation. These are then considered with respect to the latest version of usability heuristics published by Nielsen [36], leading to a set of guidelines which take account of usability and learning issues. Most importantly we claim that taking a socio-constructivist approach leads to a consideration of the interaction between usability and learning.

The range of educational software is becoming more diverse, ranging from so called generic software packages such as word processors, WWW browsers, and conferencing systems to subject specific multimedia applications. Given a socio-constructivist view of learning which maintains that educational software is defined by its use rather than abstract features, our proposed heuristics can only be applied as a predictive evaluation instrument with respect to a perceived educational context. This contextual specificity overrides the apparent general scope of generic software, making it possible legitimately to apply the heuristics to all types of educational software.

2. A socio-constructivist view of learning

In her review of theories of learning and multimedia applications, Atkins [3] suggests that learning with interactive courseware delivered on advanced technology platforms can be categorised in terms of two dominant underlying views of learning: the behaviourist and the cognitive. Within the cognitive category she distinguishes between 'weak' artificial intelligence and constructivist views of learning. While there are many interpretations of constructivism [44], the description provided by Soloway et al. [53] as learning and understanding being "active, constructive, generative processes such as assimilation, augmentation, and self-reorganisation" (p. 190) captures the essence of the constructivist perspective. Solway et al. also address the issue of the social context of learning by synthesising the work of a number of authors to state that the central notion of socioculturism is "that learning is enculturation, the process by which learners become collaborative meaning-makers among a group defined by common practices, language, beliefs, use of tools, and so on" (p. 190). Taken together, the central notions of constructivism and socioculturism can be described as socio-constructivism.

As theories of learning have developed and educationalists have gained more experience of using computer-based technology, there has been a shift of emphasis from the behaviourist paradigm, through the weak artificial intelligence approach, to a constructivist view. For most educationalists, constructivism offers far more scope for realising possible learning benefits of using information and communication technology. In fact, Reeves [47] refers to the claim by Gagne and Glaser [15] that virtually all self-respecting instructional design theorists now claim to be cognitivists.

Many writers have expressed their hope that constructivism will lead to better educational software and better learning (e.g., [5,27,40]). They stress the need for open-ended exploratory authentic learning environments in which learners can develop personally meaningful and transferable knowledge and understanding. The lead provided by these writers has resulted in the proposition of guidelines and criteria for (i) the development of constructivist software (e.g., [11,13,14,17–19,21,25,48,51,56]) and (ii) the identification of new pedagogies (e.g., [29,61]).

A recurrent theme of these guidelines, software developments and suggestions for use is that learning should be authentic. This notion of authenticity can be considered from both cognitive and contextual perspectives. The issues emerging from these perspectives will be used to inform this discussion.

2.1. Cognitive authenticity

A tenet of constructivism is that learning is a personal, idiosyncratic process, characterised by individuals developing knowledge and understanding by forming and refining concepts [43]. This leads to the notion of cognitively authentic learning experiences in which learners are assisted in some way to construct and refine concepts in personally meaningful ways. A review of the literature points to three seminal concepts which originate from the notion of cognitive authenticity: credibility, complexity and ownership.

For learners to feel that an environment offers credible opportunities for learning they need to be able to explore the behaviour of systems, environments or artefacts, e.g. simulations. The environment should provide the learner with intrinsic feedback, which represents the effects of the learner's action on the system, environment or artefact. Learners should be able to express personal ideas and opinions, with the environment providing a mechanism for the articulation of these ideas. Papert [38,40] has described computer based microworlds acting in this way as "incubators of knowledge". In addition, learners should be able to experiment with ideas and try out different solutions to problems. In this sense they should be able to adopt multiple perspectives by engaging in activities which support multiple knowledge representations, experience varied cases and contexts, and have varied purposes for knowledge [21]. Using analogies and extension [10] and constructionism [22] are relevant here. Ainsworth et al. [1] report two evaluation studies which show that children as young as six years can benefit from the use of multi-representational software.

Grabinger and Dunlap [17] emphasise that learners should be presented with authentic environments, which are representative of interesting and motivating tasks, rather than contrived sterile problems. Learners may need help in coping with complexity. Strategies to help learners include scaffolding [53], anchoring [10] and problem based environments [19,58].

A sense of ownership should be a prominent feature of learning. Learners need to be encouraged to take responsibility for learning. Strategies for encouraging metacognition and intentional learning are relevant here [52].

2.2. Contextual authenticity

It is now commonly advocated that cognition and learning are situated in specific learning contexts (e.g. [5,9,30,42]). A situated view of learning implies that effects on learning of using information and communication technology will depend on the context in which it is used, with all the components of a learning environment (people and artefacts) interacting and contributing to the learning process. Some writers, e.g. Pea [42], see ‘intelligence’ in a given context as distributed between people and information and communication technology applications. Clearly a critical feature of any learning environment will be the role played by the teacher, and many educationalists now believe that a very important role for educational software is to foster a move from teacher centred to learner-centred pedagogies.

One way in which the teacher’s role is important is in guiding students to appropriate contexts, including appropriate selection of educational software. This is especially significant when, as is often the case, it is necessary for students to go beyond understanding a concept in a specific context, so that they can apply it more generally. It can be argued that “the point of an academic education is that knowledge has to be abstracted, and represented formally, in order to become generalisable and therefore more generally useful” ([30], p. 19–20). In such cases it is clear that “situated cognition in the context of education is not concerned simply with learning about the world, but with learning about a way of looking at the world” ([30], p. 20).

A belief in contextualised learning also emphasises collaborative learning in which peer group discussion and work is prominent in helping students to learn (e.g., [26,60]). In this context the role of teacher will change to a manager and facilitator of learning, rather than a director [54]. Clearly another important aspect of the learning context is the curriculum.

3. Established approaches to predictive evaluation

Checklists of questions which attempt to deal with both learning and usability issues date back to the early days of educational software use (e.g., [23]). They are still popular (e.g., [50]), with new lists appearing for current software environments such as CD-ROM based applications [57] and hypertext software [59]. The ability of checklists to predict educational issues in all but a naive and superficial way has been questioned by several researchers (e.g., [31]). McDougall and Squires cite a number of authors who identify problems which evaluators have found with the use of checklists as predictive evaluation tools:

- it is difficult to indicate relative weightings for questions [63]
- selection amongst educational software of the same type emphasises similarities rather than differences [54]
- the focus is on technical rather than educational issues [37]
- it is not possible to cope with the evaluation of innovative software [24]
- it is not possible to allow for different teaching strategies [63]
- off-computer, teacher generated uses are not considered [54]
- evaluation in different subject areas requires different sets of selection criteria [28]

Squires and McDougall [54] maintain that these problems are symptomatic of the failure to adopt a situated perspective on the use of educational software. They suggest that the first four problems stem from a focus on the software application as an object of evaluation in its own right rather than the evaluation of its use, i.e. the use of the software is not conceived in a distributed fashion. In their opinion the next two problems indicate that the diversity and complexity of the classroom, and the teacher's role in managing this complexity, do not feature in the design of checklists. They claim that the last problem again indicates a non-situated perspective—generalised notions of good practice in a subject discipline are employed, rather than issues relating to specific educational situations.

Increasing dissatisfaction with checklists has led to attempts to develop instruments which address the problems inherent in the checklist paradigm. For example, the California Instructional Technology Clearing House [6] has produced guidelines for the evaluation of educational software in a 'rubric format' which is a "major departure from the checklists developed in prior years" (p. 1). About 100 screening criteria are provided which must be met before a program is accepted for evaluation. The rubrics deal with the following areas: California Curriculum Content, Instructional Design for Learners, Program Design, Assessment, and Instructional Support Materials. Within each area the rubrics are presented as a matrix with three columns (makes an excellent, good or minimal case for recommendation) matched against rows corresponding to desirable features. Supplementary rubrics are provided for distance learning resources, on-line learning experience, presentation tools, reference tools, and productivity tools.

These rubrics attempt to provide a comprehensive framework for evaluators, as illustrated by one of the instructional design rubrics which requires a program to feature a variety of creative teaching and learning approaches such as: constructivist approaches, co-operative learning groups, collaboration with other learners via a network or the Internet, strategies to encourage multiple intelligences and a variety of learning styles, independent investigations, open-ended questioning, and strategies to stimulate student creativity. However, while a serious attempt has been made to provide an holistic view of the use of software in the classroom, the result is much like a very extensive checklist. As with so many evaluation instruments, an attempt to capture contextual sensitivity has resulted in an excessively fine grained approach which is tedious to apply.

4. An heuristic approach and predictive evaluation

Techniques for evaluating usability are a core concern for interaction design and a much greater variety of techniques have been developed to evaluate commercial software than for educational software. There are techniques which are appropriate for all stages of the design process, for predictive, formative and summative evaluation. Predictive evaluation techniques include heuristic evaluation, walkthroughs and modelling [45]. Of these, euristic evaluation [36] appears most suitable for teachers to predictively evaluate educational software. Most forms of walkthroughs developed for HCI are strongly cognitively oriented and require very detailed task knowledge, which is feasible for evaluating software designed to promote clearly defined skill development but not for software which encourages creativity and which can be used in different ways by different students. Modelling approaches, e.g. the GOMS technique developed by Card et al. [7], are typically too fine grained to be of practical use to teacher evaluators.

Any technique used by teachers needs to be relatively quick and easy to use. Heuristic evaluation is designed to address key usability issues in a cost effective way. High level guidelines or heuristics focus reviewers' attention as they work their way through the system, using their expertise to role-play the behaviour of a typical user. The latest version of usability heuristics published by Nielsen [36, p. 30] are as follows:

- *Visibility of system status*: the system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
- *Match between the system and the real world*: the system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system oriented terms. Follow real world conventions, making information appear in a natural and logical order.
- *User control and freedom*: users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
- *Consistency and standards*: users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
- *Error prevention*: even better than good error messages is a careful design which prevents a problem from occurring in the first place.
- *Recognition rather than recall*: make objects, actions and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
- *Flexibility and efficiency of use*: accelerators—unseen by the novice user—may often speed up the interaction for the expert user to such an extent that the system can cater for both inexperienced and experienced users. Allow users to tailor frequent actions.
- *Aesthetic and minimalist design*: dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
- *Help users recognise, diagnose, and recover from errors*: error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

- *Help and documentation*: even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Normally each evaluator does two or more passes through the interface with the aim of inspecting the flow of the interface from screen to screen, and the specific features of each individual screen, such as dialogue boxes, feedback messages, etc. Research has shown that the use of these heuristics by five expert valuators will typically lead to the identification of about 75% of the design problems associated with a package [35]. Thus, in HCI evaluation the application of a carefully selected set of heuristics by a group of experts can lead to a principled but cost effective evaluation methodology. We suggest that this approach can be adopted in predictive educational software evaluation.

5. Usability heuristics and a socio-constructivist view of learning

In this section the concepts arising from the notions of cognitive and contextual authenticity will be used as a basis for relating Nielsen's usability heuristics to socio-constructivist criteria for learning. A review of some of the possible relationships between these heuristics and learning criteria is given in Table 1.

An inspection of Table 1 shows that there is an interaction between at least one of the heuristics and each of the concepts of credibility, complexity and ownership, suggesting a relationship between cognitive complexity and Nielsen's heuristics. Although there is a good basis for many of the interactions shown in the table, some are less strongly grounded and need to be validated empirically. Some examples of possible inter-related issues have been identified for 19 of the possible 50 areas of interaction. The potential relationships between the heuristics and the contextual concepts of collaboration, curriculum and ownership appear to be less obvious than for credibility and complexity.

5.1. Credibility

There is scope for a strong relationship between credibility and the Nielsen heuristics. Examples of five possible inter-related issues arising from the ten possible interactions are presented.

5.1.1. Feedback and designer/learner models

The design of a computer based learning environment will incorporate decisions about how learning tasks will be presented to a learner, i.e. a computational learning environment will be defined in terms of the designer's mental model. As the learner interacts with an environment the system will provide intrinsic feedback determined by the way in which the designer's model works. This is consistent with the constructivist view that by active engagement with an educational task, learners receive feedback on their performance. Problems will arise if the learner does not share the same mental model as the designer, or is at least aware that the models may be different. This is a complex issue as problems can

Table 1
The relationship between usability and learning evaluation heuristics

	Cognitive authenticity			Contextual authenticity	
	Credibility	Complexity	Ownership	Collaboration	Curriculum
System status visibility	Feedback and designer/learner models	Navigation			
Match system/world	Cosmetic authenticity	Representations of the real world			
User control			Learner control	Shared responsibility	
Consistency		Symbolic representation		Consistent protocols	Subject content
Error prevention		Peripheral cognitive errors			
Recognition	Representational forms				
Flexibility	Multiple views/ representations		Tailoring the interface		Teacher customisation
Aesthetic design		Superficial complexity			
Error recovery	Interaction flow	Pedagogical techniques	Metacognition		
Help/ documentation		Learners' support materials			

occur at different levels of system definition. Squires [55] reports problems at the interface level when learners used an application based on the direct manipulation of a ‘datacube’ consisting of a set of serially arranged datasheets. Learners were confused by the way the program operated as the designer’s model assumed data values to be defined with respect to datasheets while they typically assumed the values to be defined with respect to the datacube. The debate over whether local or global scoping is appropriate in the design of Logo and Boxer [12] illustrates the importance of this issue at the lower system level of interpreter design.

5.1.2. Cosmetic authenticity

Many designers find it difficult to resist the using elaborate multimedia features and may use them gratuitously, often producing a superficial match with the real world that presents the learner with convincing images of reality. In terms of learning, while this level of authenticity may be attractive and motivating, it does not necessarily have any intrinsic benefits for learning—it may even be misleading. For example, there is a longstanding convention that vessels carrying oxygenated blood are shown in red and those carrying deoxygenated blood in blue. Changing this colour coding for cosmetic reasons would be confusing and detrimental to learning.

5.1.3. Representational forms

Learners should not be burdened with having to learn and remember arcane forms of interaction; manipulating educational software should not compromise the learning experience. In this sense the interface should place a low cognitive demand on the learner and functionality should be obvious. Symbols, icons and names should be intuitive within the context of the learning task. Being able to give meaningful variable names in Logo, e.g. ‘side’ for the length of a square rather than ‘x’, is an example of stressing recognition rather than recall. In summary, representational forms should be both intuitive and meaningful.

5.1.4. Multiple views/representations

The idiosyncratic nature of learning implied by constructivism means that it is important to be able to provide multiple representations and views within a learning environment. In this sense educational software needs to be flexible and support a variety of representations, e.g. the Envisioning Machine [49] which presents parallel views of motion corresponding to ‘observable’ and ‘Newtonian’ worlds. Different forms of media can be used both independently and in combinations to offer different perspectives; in fact this is a defining rationale for educational multimedia. In addition software can be designed to support different strategies for learning, e.g. serialist and holistic learners [41].

5.1.5. Interaction flow

Users appreciate a smooth flow of interaction with an application. In educational applications it is common practice to provide extrinsic feedback to the learner in the form of error messages, hints and ‘bolt-on’ tutorial sequences. Alty [2] has noted that such approaches can interrupt the flow of interaction. His research showed that too frequent error messages stopped students concentrating on the learning task in hand. Clearly there is a need to balance the sometimes conflicting demands of smooth interaction flow and extrinsic feedback.

5.2. Complexity

The correlation between complexity and the Nielsen heuristics is also strong with seven potential inter-related issues arising from the ten possible interactions.

5.2.1. Navigation

Learning environments need to be navigable in two senses. First, a basic requirement is that past, present and future possibilities for manipulating an application should be evident from the interface design. Second, there should be some representation which learners can use to assess how these possibilities relate to their task requirements. There are well established interface techniques designed to support ‘manipulative navigation’, e.g. menus, dialogues and hypermedia structures, but task focused navigation is more problematic. A simplistic approach to task focused navigation is to provide extrinsic rewards which are intended to motivate the learner to follow an intended route through a program. This approach is clearly incompatible with a constructivist view of learning. A more appropriate approach from a constructivist perspective is to rely on intrinsic feedback, i.e. feedback produced by the program in response to a learner’s attempts to complete a learning task. The implication here is that the interface design developed to afford manipulative possibilities should also be capable of providing appropriate intrinsic feedback.

5.2.2. Representations of the real world

Usability design is often predicated on making the interface analogous to a representation of some aspect of the real world context, e.g. the Macintosh desk-top metaphor. Grudin [20] refers to this approach as "external analogic or metaphoric correspondence of the design to the world beyond the computer domain". Educationalists have also predicated design from a learning perspective on a correspondence with the real world: educational simulations which attempt to represent real world systems and processes are common, many claims are made for the cognitive benefits of using computer based modelling systems, and computer based microworlds have been described as:

> A subset of reality or a constructed reality whose structure matches that of a given cognitive mechanism so as to provide an environment where the latter can operate effectively. These concepts lead to the project of inventing microworlds so structured as to allow a human learner to exercise powerful ideas or intellectual skills.

([39], p. 204)

When interface analogies are used to represent complex learning tasks the idiosyncratic views of the world held by learners may be at odds with the designer's views. Thus, the use of a real world representation to provide both an interface design rationale and an exploratory and expressive domain for learners may be a source of confusion for learners. With careful design there is the possibility of creating powerful synergistic learning environments with interaction contexts matching learning aims. However, there is also a possibility for a mismatch between the interaction context and learning aims, leading to confusing and misleading learning environments.

5.2.3. Symbolic representation and terminology

The use of interface artefacts such as icons, symbols and widgets is now a firmly established aspect of interface design. To assist learners cope with complexity, the same symbols, icons and names are often used to represent educational 'objects'. Concepts and terminology should be used consistently throughout an application, and where possible they should be used consistently across applications. Learners who explore the web may find this a problem, but within the design of a single educational resource, teachers should check that designers have been consistent. In addition, attention needs to be paid to implicit design assumptions underlying the use of artefacts. For example, the use of 'sliders' to allow users to change the value of variables presents an intuitive and easy to use interface. There is an implicit assumption that the variable being changed is continuous, but will learners be aware of this?

5.2.4. Peripheral cognitive errors

Errors arise from two sources—usability errors originate from manipulating software and cognitive learning errors arise when concepts are developed and refined in an attempt to understand the complexity of a learning environment. It is clear that learners need to be protected from usability errors. However, in constructivist terms learners need to make and remedy cognitive errors as they learn. Thus, it is necessary to distinguish between errors which compromise efficient manipulation of an application and those arising from developing and refining concepts. Cognitive errors should be relevant to the major learning issues. It is annoying and distracting to make ‘peripheral’ errors related to usability problems, such as putting in impossible values for physical parameters. Such errors can be anticipated and avoided, e.g. allow the selection of a variable’s value by moving a slider which has fixed maximum and minimum values or provide ‘novice versions’ of an application which have pre-set default values.

5.2.5. Superficial complexity

A multimedia collage of text, graphics, video and sound may be attractive. However, it may be the case that complex presentation is being used to present very limited educational material. In this sense the environment is only complex in a superficial sense. A deeper complexity is one that relates the structure of the multimedia environment to conceptual development, and navigating this cognitive structure becomes a learning experience in its own right. In superficially complex environments there will be a heavy cognitive load imposed by navigating the interface. The well known expression ‘lost in hyperspace’ comes to mind in this context. In environments with ‘navigational fidelity’ the structure of the environment relates directly to cognitive tasks.

5.2.6. Pedagogical techniques

A tenet of constructivism is that people learn by their mistakes. By implication learners need access to rich and complex environments where they can express their own ideas and explore different solutions to problems. While the provision of rich complex environments is consistent with a constructivist approach, it is clear that learners will typically require help to recover from cognitive errors as they work in these learning environments. As noted before, well established techniques have been developed to cope with complexity, such as scaffolding, bridging, anchoring, problem based environments and cognitive conflict, which provide design pointers for the development of complex environments which are genuinely usable by learners. Soloway et al. [53] provide an example of the use of one of these techniques. In the Model-It software, scaffolding provides the rationale for the provision of a set of pre-defined high-level objects (e.g. stream, bugs, golf course) which can be used by students to explore environmental planning issues.

5.2.7. Learners’ support materials

If learners are presented with very complex environments which present unmitigated representations of real world complexity they are unlikely to learn efficiently, if at all. Various techniques have emerged which attempt to mitigate complexity. For example, some simulations allow restricted versions to be run by limiting the number of variables that can be changed, e.g. a simulation of the manufacture of sulphuric acid depending on input values for temperature, pressure, the ratio of sulphur dioxide and oxygen, and the presence of a catalyst could be presented as a simplified version which only allowed students to vary the pressure and temperature. Other educational applications provide default input values for the novice user. Providing specific problems and tasks, often as extra documentation, can help to focus learners’ attention.

5.3. Ownership

Ownership of ideas is an important part of learning successfully. Three issues arise from the ten possible interactions.

5.3.1. Learner control and self-directed learning

A tenet of constructivism is that learners direct their own learning either individually or through collaborative experiences. This implies that learners need to find their own pathways through learning; a philosophy that underpins hypertext and many web-based instructional systems [34]. E-mail, listservs and web browsers also support this approach by enabling students to search for information and discuss issues with others across the world. In fact, the notion of learner control has been well established in the educational computing literature for some time (e.g., [4,8,16,32,62]). The use of software, which provides high levels of user control will help students feel that they are instrumental in determining the pattern and process of the learning experience, i.e. in developing a sense of ownership.

5.3.2. Tailoring the interface

Being able to tailor the interface to support an individual learner's needs and learning strategies will become increasingly important as learners become more sophisticated computer users. As a focus on individual learning in the home increases, the need for bespoke versions of software will increase. This aspect of flexibility is set to become more and more important.

5.3.3. Metacognition

Metacognition, in which learners reflect on their own cognition to improve their learning, has been advocated by a number of writers, e.g. Scardamalia et al. [52] and Papert [39]. It is claimed that by a conscious personal appraisal of a learner's cognitive processes, individuals can improve their capacity to learn. Clearly, for this to be effective there is an assumption that learners feel a sense of ownership of their learning.

5.4. Collaboration

Collaboration is a central tenet of socio-constructivism, which is supported by computer networks. However, collaboration is a relatively new concept in education. Collaboration inter-relates on only two occasions, but in important ways, with Nielsen's heuristics. Further use of the heuristics may reveal additional interactions.

5.4.1. Shared responsibility

The notion of peer group learning implicit in a socio-constructivist view introduces a social dimension in which learners delegate, to some extent, the control of the learning experience to other members of the group and to the group dynamic as a whole [26]. There is also an implication that with the autonomy that computer based learning environments can bring to the learner, there will be a shift in the balance of responsibility from the teacher to the learner. In this sense, the teacher will become more of a facilitator and manager of the learning process, rather than a director of the process.

5.4.2. Consistent protocols

Collaborative working and learning assumes shared resources, documents and other artefacts across a network of users and their environments. For this to be effective, learners need to develop rules to guide their working practices.

5.5. Curriculum

Two examples of inter-related issues arise out of the ten possible areas of interaction between the concept of curriculum and the Nielsen heuristics.

5.5.1. Subject content

The intended scope of learning is defined by the curriculum. In this sense the knowledge base of an educational application needs to correspond to some extent to the curriculum in question. This is relatively easy to assess for a subject specific application. Making a similar assessment for a generic package, such as a spreadsheet, will be more problematic. In cases such as these the assessment will be in terms of the skills the application supports.

5.5.2. Teacher customisation

Teachers will often feel the need to adapt software to the specific needs of their students or to match particular curriculum requirements. Thus, a good feature of an educational application is the facility for the teacher to customise the application. In the case of generic applications this may consist of such things as defining procedures with sub-procedures in programming languages, developing macro models with component sub-parts in modelling systems, and creating macros in spreadsheets.

6. Learning with software heuristics

The analysis in the previous section of the possible relationships between the Nielsen heuristics and the notions of cognitive and contextual authenticity can be used as the basis of developing a set of ‘learning with software’ heuristics. By looking for links between the individual relationships it is possible to propose an initial set of these heuristics:

- A need for a *match between designer and learner models* is implied by considering intrinsic feedback and the relationship between learner and designer models. At some level of system definition, intrinsic feedback should provide a legitimate and understandable representation of cognitive tasks which ensures that the model formed by learners will be consistent with the designer’s model. A low level representation close to the core language of a system will probably be confusing. Representation at too high a level will result in a superficial model, which will not be of genuine use, and may even be misleading. The designer and learner models do not need to be the same, but there should be no differences between them, which would cause misconceptions in the learner's model of the relationship between the interface and the system.

- A requirement for *navigational fidelity* is apparent when navigational structure, cosmetic authenticity, limited representation of the world and superficial complexity are considered. Interface designs that provide good usability may compromise authenticity by providing simplistic representations of the real world. The use of elaborate multimedia features may result in superficially complex interfaces which focus interaction on incidental navigation, rather than intended learning tasks.
- The need to consider *appropriate levels of learner control* follows from a consideration of learner control and shared responsibility, self directed learning, tailoring and consistent protocols. A socio-constructivist view emphasises that learners should have a sense of ownership of their learning. This implies that they should have the maximum amount of control while still working in a supportive learning environment. The locus of control among peers during collaborative work is important.
- The need for the *prevention of peripheral cognitive errors* is implied by the relationship between complexity and error prevention. Cognitive errors should be relevant to the major learning issues. Peripheral usability related errors should be anticipated and avoided. Where possible, novice versions of an application should be provided.
- The requirement for *understandable and meaningful symbolic representation* follows from a consideration of representational forms and the use of symbols within and across applications. Learners should not be burdened with having to learn and remember arcane forms of interaction. The interface should place a low cognitive demand on the learner and functionality should be obvious. The same symbols, icons and names used to represent educational 'objects' and concepts should be used consistently throughout an application.
- The need to *support personally significant approaches to learning* follows from a consideration of multiple representations, learners' support materials and metacognition. It should be clear what learning styles are supported and which aspects of an application's design relate to learning style characteristics.
- The need for *strategies for the cognitive error recognition, diagnosis and recovery cycle* is implicit from the discussion of pedagogical techniques. Established strategies to promote the cycle of recognition, diagnosis and recovery should be used, e.g. cognitive conflict, scaffolding, and bridging.
- That there is a clear need for a *match with the curriculum* is evident from a consideration of curriculum relevance and teacher customisation.

The heuristics proposed in this section provide a principled 'first cut' at developing a predictive evaluation tool that takes account of both usability and learning issues. By addressing a socio-constructive vision of learning we believe that the heuristics cope with the integration of usability and learning. There is clearly a need for a large-scale empirical study to test the efficacy of these heuristics; a study which mirrors in an educational context the factor analysis of Nielsen [36], which will form the next stage of our work.

References

[1] S.E. Ainsworth, P.A. Bibby, D.J. Wood, Information Technology and multiple representations: new opportunities—new problems, Journal of Information Technology for Teacher Education 6 (1) (1997) 93–104.
[2] J. Alty, Computers and Graphics 17 (3) (1993) 203–218.
[3] M.J. Atkins, Theories of learning and multimedia applications: an overview, Research Papers in Education 8 (2) (1993) 251–271.
[4] D. Blease, Evaluating Educational Software. Croom Helm, London, 1988.
[5] J.S. Brown, A. Collins, P. Duguid, Situated cognition and the culture of learning, Educational Researcher 18 (1989) 32–42.
[6] California Instructional Technology Clearing House, Guidelines for the Evaluation of Instructional Technology Resources for California Schools. California Instructional Technology Clearing House, Modesto, CA, 1997.
[7] S.K. Card, T.P. Moran, A. Newell, The Psychology of Human Computer Interaction. Lawrence Erlbaum Associates, Hillsdale, NJ, 1983.
[8] D. Chandler, Young Learners and the Microcomputer. Open University Press, Milton Keynes, UK, 1984.
[9] W.J. Clancey, Situated cognition: how representations are created and given meaning, in: R. Lewis, P. Mendelsohn (Eds.), Lessons From Learning, North-Holland, Amsterdam, 1994, pp. 231–242.
[10] Cognition and Technology Group at Vanderbilt, Emerging technologies, ISD, and learning environments: Critical perspectives, Educational Technology Research and Development 40 (1) (1992) 65–80.
[11] D. Cunningham, T.M. Duffy, R. Knuth, Textbook of the future, in: C. McKnight (Ed.), Hypertext: A Psychological Perspective, Ellis Horwood, London, 1993.
[12] A.A. diSessa, A principled design for an integrated computational environment, Human–Comptuer Interaction 1 (1985) 1–47.
[13] M.P. Driscoll, Psychology of Learning for Instruction, Allyn and Bacon, Boston, MA, 1994.
[14] T.M. Duffy, D.J. Cunningham, Constructivism: Implications for the design and delivery of instruction, in: D.H. Jonassen (Ed.), Handbook of Research for Educational Communications and Technology, Macmillan Library Reference USA, New York, 1996.
[15] R.M. Gange, R. Glaser, Foundations in learning research, in: R.M. Gange (Ed.), Instructional Technology Foundations, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, pp. 49–83.
[16] D. Goforth, Learner control = decision making + information: A model and meta-analysis, Journal of Educational Computing Research 11 (1) (1994) 1–26.
[17] R. Grabinger, J.C. Dunlap, Rich environments for active learning: A definition, Association for Learning Technology Journal 2 (3) (1995) 5–34.
[18] R.S. Grabinger, Rich environments for active learning, in: D.H. Jonassen (Ed.), Handbook of Research for Educational Communications and Technology, Macmillan Library Reference USA, New York, 1996.
[19] R.S. Grabinger, J.C. Dunlap, J.A. Duffield, Rich environments for active learning in action: problem-based learning, Association for Learning Technology Journal 5 (2) (1997) 5–17.
[20] J. Grudin, The case against user interface consistency, Communications of the ACM 32 (10) (1989) 1164–1173.
[21] M.J. Hannafin, S.M. Land, The foundations and assumptions of technology-enhanced student centres learning environments, Instructional Science 25 (1997) 167–202.
[22] I. Harel, S. Papert, Software design as a learning environment, in: H.I. Papert, S. Papert (Eds.), Constructionism, Ablex, Norwood, NJ, 1991.
[23] W. Heck, J. Johnson, R. Kansky, Guidelines for Evaluating Computerised Instructional Materials, National Council for Teachers of Mathematics, Reston, VA, 1981.
[24] R. Heller, Evaluating software: A review of the options, Computers and Education 17 (4) (1991) 285–291.
[25] P.C. Honebein, T.M. Duffy, B.J. Fishman, Constructivism and the design of authentic learning environments: context and authentic activities for learning, in: T.M. Duffy, J. Lowyck, D.H. Jonassen (Eds.), Designing Environments for Constructive Learning, Springer-Verlag, Berlin, 1993, pp. 87–108.
[26] C. Hoyles, C. Healy, S. Pozzi, Groupwork with computers: An overview of findings, Journal of Computer Assisted Learning 10 (4) (1994) 202–215.

[27] D.H. Jonassen, Thinking technology: Toward a constructivist design model, Educational Technology 34 (3) (1994) 34–37.
[28] P.K. Komoski, Educational microcomputer software evaluation, in: J. Moonen, T. Plomp (Eds.), Eurit86: Developments in Educational Software and Courseware, Pergamon Press, Oxford, 1987, pp. 399–404.
[29] D.M. Laurillard, Learning through collaborative computer simulations, British Journal of Educational Technology 2 (1992) 164–171.
[30] D.M. Laurillard, Rethinking University Teaching: A Framework for Effective Use of Educational Technology, Routledge, London, 1993.
[31] A. McDougall, D. Squires, A critical examination of the checklist approach in software selection, Journal of Educational Computing Research 12 (3) (1995) 263–274.
[32] A. McDougall, D. Squires, Student control in computer based learning environments, in: A.D. Salvas, C. Dowling (Eds.), Computers and Education: On the Crest of a Wave, Proceedings of the Fourth Australian Computer Education Conference, Computer Education Group of Victoria, Melbourne, 1986, pp. 269–272.
[33] R. Molich, J. Nielsen, Improving a human–computer dialogue, Communications of the ACM 33 (3) (1990) 338–348.
[34] J.H. Murray, Hamlet on the Holodeck: The Future of Narrative in Cyberspace, The Free Press, New York, 1997.
[35] J. Nielsen, Finding usability problems through heuristic evaluation, in: P. Bauersfield, J. Bennet, G. Lynch (Eds.), Human Factors in Computing Systems, CHI'92 Conference Proceedings, Academic Press, New York, 1992, pp. 373–380.
[36] J. Nielsen, Usability inspection methods, in: J. Nielsen, R.L. Mack (Eds.), Usability Inspection Methods, John Wiley, New York, 1994, p. 30.
[37] Office of Technology Assessment, Power On! New Tools for Teaching and Learning. US Government Printing Office, Washington, DC, 1988.
[38] S. Papert, Mindstorms: Children, Computers and Powerful Ideas, Basic Books, New York, 1980.
[39] S. Papert, Computer-based microworlds as incubators for powerful ideas, in: R.P. Taylor (Ed.), The Computer in the School: Tutor, Tool, Tutee, Teachers College Press, New York, 1980.
[40] S. Papert, The Children's Machine: Rethinking School in the Age of the Computer, Basic Books, New York, 1993.
[41] G. Pask, Conversational techniques in the study and practice of education, British Journal of Educational Psychology 46 (1976) 12–25.
[42] R. Pea, Practices of distributed intelligence and designs for education, in: G. Salomon (Ed.), Distributed Cognitions: Psychological and Educational Considerations, Cambridge University Press, Cambridge, 1993, pp. 47–87.
[43] J. Piaget, The Origins of Intelligence in Children, International University Press, New York, 1952.
[44] D.C. Phillips, The good, the bad, the ugly: The many faces of constructivism, Educational Researcher 24 (7) (1995) 5–12.
[45] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, T. Carey, Human–Computer Interaction, Addison-Wesley, Wokingham, UK, 1994.
[46] J. Ramsay, A. Barabesi, J. Preece, Informal communication is about sharing objects in media, Interacting with Computers 6 (3) (1996) 277–283.
[47] T.C. Reeves, Evaluating what really matters in computer-based education, in: M. Wild, D. Kirkpatrick (Eds.), Computer Education: New Perspectives, Edith Cowan University Press, Perth, Australia, 1994, pp. 219–246.
[48] L.P. Rieber, Computer-based microworlds: A bridge between constructivism and direct instruction, Educational Technology Research and Development 40 (1) (1992) 93–106.
[49] J. Roschelle, Designing for cognitive communication: Epistemic fidelity or mediating collaborative enquiry, in: D.L. Day, D.K. Kovacs (Eds.), Computers, Communication and Mental Models, Taylor and Francis, London, 1996, pp. 15–27.
[50] J.E. Rowley, Selection and evaluation of software, Aslib Proceedings 45 (3) (1993) 77–81.
[51] J.R. Savery, T.M. Duffy, Problem based learning: An instructional model and its constructivist framework, Educational Technology 35 (5) (1995) 31–38.
[52] M. Scardamalia, C. Bereiter, R. McLean, J. Swallow, E. Woodruff, Computer supported intentional learning environments, Journal of Educational Computing Research 5 (1989) 51–68.

[53] E. Soloway, S.L. Jackson, J. Klein, C. Quintana, J. Reed, J. Spitulnik, S.J. Stratford, S. Studer, J. Eng, N. Scala, Learning theory in practice: case studies in learner-centred design, in: R. Mack, L. Marks, D. Collins (Eds.), CHI'96 Proceedings, ACM, 1996, pp. 189–196.
[54] D. Squires, A. McDougall, Choosing and Using Educational Software: A Teachers' Guide, Falmer Press, London, 1994.
[55] D. Squires, The use of direct manipulation in educational software design, in: D. Tinsley, T. van Weert (Eds.), Liberating the Learner, Fourth World Conference on Computers in Education, Chapman and Hall, Birmingham, 1995, pp. 273–281.
[56] D. Squires, Can multimedia support constructivist learning?, Teaching Review 4 (2) (1996) 10–17.
[57] S. Steadman, C. Nash, M. Eurat, CD-ROM in Schools Scheme Evaluation Report, National Council for Educational Technology, Coventry, UK, 1992.
[58] K. Tobin, G. Dawson, Constraints to curriculum reform: Teachers and the myths of schooling, Educational Technology Research and Development 40 (1) (1992) 81–92.
[59] D. Tolhurst, A checklist for evaluating content-based hypertext computer software, Educational Technology 32 (3) (1992) 17–21.
[60] D. Watson, A. Moore, V. Rhodes, Case studies, in: D. Watson (Ed.), The Impact Report, King's College London, London, 1993, pp. 61–96.
[61] D.M. Watson, The computer in the social science curriculum, in: T. Plomp, J. Moonen (Eds.), The International Journal for Educational Research 17 (1) (1992).
[62] J.J. Wellington, Children, Computers and the Curriculum, Harper and Row, New York, 1985.
[63] J. Winship, Software review or evaluation: Are they both roses or is one a lemon? Paper presented at the Proceedings of the Australian Computer Education Conference, Perth, 1988.