The Amateur Creator

Stephen Boyd Davis
Lansdown Centre for Electronic Arts
Middlesex University
Cat Hill, Barnet, Herts, UK
s.boyd-davis@mdx.ac.uk

Magnus Moar
Lansdown Centre for Electronic Arts
Middlesex University
Cat Hill, Barnet, Herts, UK
m.moar@mdx.ac.uk

ABSTRACT
Important design problems are raised in developing software for amateur users, a group distinguished here from novices. The authors argue that these design problems can be approached by understanding how systems for amateurs are derived from those for skilled users, through a combination of transformations we describe as foregrounding, backgrounding, automation, integration and constraining. Useful comparisons are offered with popular product designs. A broader, partly historical, context is then described in which media technologies propagate from use by specialists to use by these amateurs, and the latter change from consumers to creators. The discussion is focused by a description of difficulties with existing software encountered in the course of a creative schools-based project, intended to enable young users both to explore virtual worlds and to design and populate them with their own avatars. The authors argue that HCI design would benefit from a clearer grasp of the special characteristics of designing for amateur users and of transforming existing software for their use.

Author Keywords

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
Our past four years’ work with young school learners building shared three-dimensional worlds has made us strongly aware of difficulties with existing software, both when individual packages are considered independently and when they are considered as parts of an overall system for creative work. In VERTEX (http://www.vertex.mdx.ac.uk/), our particular interest has been in enabling young schoolchildren (9 to 11 years) not just to explore shared virtual worlds, but also to create worlds and the objects and characters within them. We used inexpensive desktop net-based virtual worlds software, helping both teachers and children to investigate the creative learning possibilities. The starting point for VERTEX was the importance of learning through making – hence our interest in technologies which enable amateurs to create. (Amateurs are not quite the same as novices, since, as discussed below, they are not in transition towards being an expert). Taking the constructionist idea that learning can occur deeply and effectively through the making of personally meaningful artefacts and projects [26], and the concept that imagination and play are fundamental factors in learning [37], VERTEX placed children in the role of designers and producers, as well as users, of these media. In doing so, the project aimed to observe the learning that might arise through children’s direct participation in these processes. Such learning outcomes, which were broadly successful, are described elsewhere [18], [2].

In attempting to make sense here of the difficulties encountered, we draw on two existing bodies of work. One deals with the relationship between computer tools and human creative activity (for which we offer a relevant definition). But we also consider a less obvious approach. Product design has a long history of catering for amateur users, including amateurs engaged in creative activity. We relate this to the propagation of media technologies, from being used by specialists on behalf of consumers, to use by a majority who are both consumers and creators. This consumer-creator lies at the confluence of classical HCI theory, with its traditional concerns with tools for making and consuming information, and newer approaches concerned with issues such as pleasure and engagement [28], [5] highlighted in Monk’s keynote to HCI 2002 [19]. The discussion is intended to assist the HCI community in thinking about the special characteristics of design for all.

THE CONSUMER-CREATOR
Media technologies tend to propagate from use by specialists to use by ‘ordinary people’. This change is often accompanied – for those ordinary people – by a change in role from consumer to consumer-creator. Previous examples of propagation from the specialist to everyman
include text – writing was originally confined to professional scribes [24]; printing, which allowed far wider access to an increasing range of texts [10]; the camera, migrating from an intricate and complex pastime of the leisured classes to a media technology for everyone [11]; the computer, from use by laboratory specialists to ubiquity [25]. It can be seen that these propagations also tend to encompass a change from consuming to creating. Literacy turns text into something to make, as well as something to consume; the growth of DTP made publishers out of former consumers, and the Web enabled them to publish to the world; the Box Brownie gave people the tools to make photographs, not just enjoy the work of professionals; the home computer allowed its owner to create digital content of many kinds – printable documents, programs, games, music; amateur musicians publish their own CDs; programs like Apple’s i-Movie seem likely to make filmmakers of people who previously could only be film viewers.

It is hardly a surprise that a new media technology is at first only accessible to specialists. The devices for authoring in new technologies are often only half formed and have undergone none of the processes of transformation to access by amateurs which we describe below. In some cases the mere difficulty of implementation is a sufficient barrier. Berners-Lee describes how his original prototype browser integrated the reading and authoring of Web pages, but the difficulty of porting the tool from his NeXT machine to the many different machines on which it would be needed meant that only the browsing part was implemented [4].

**Computers and Creative Activity**

‘Create’, ‘creative’ and ‘creativity’ are contentious words, so it is important to indicate what we mean here, especially since much has been written on the different aspects of computers and creative activity. Creativity has been characterised as comprising particular distinctive procedures which computing might promote or support such as ‘association, combination, abstraction, selection, integration, and adaptation’ [14], and systems have been offered which aim to prompt creativity by provoking human insight [27]. The questions around creativity by computers [17] have been rehearsed since Babbage and Lovelace [39]. Shneiderman proposes an agenda of supportive technologies to facilitate eight actions which he defines as components of the human creative process, with a particular emphasis on an overall characteristic of fluidity and directness [32]. This is an important goal also for Sedivy & Johnson [29] in designing multimodal interaction for creating graphics, and for Terry and Mynatt’s attempts to overcome the linear non-reflexive model of the design process assumed by many existing interfaces [34].

In several of Shneiderman’s recommendations there is an overlap with our own interests: the need for better integration between tools, platforms and types of media and between local-personal and distance-collaborative modes of work; tools to support rapid externalisation, experimentation and reflection; and in particular the overall goal of greater directness and fluidity [32]. However, we should also distinguish our agenda from Shneiderman’s. For one thing our definition of creativity is simple and, we hope, non-contentious. Whereas Shneiderman defines creative work as something superior to ‘competent original work’ (p118), we use the term to mean any activity which leads to the making of an artefact not produced merely by following a set of predefined procedures (as a factory worker or a mechanism might do). In the VERTEX project, the activity of a child in designing and making an avatar to represent themselves is thus a creative act. Like Shneiderman and others we are interested in tools which offer smooth integration – a diminution of barriers between tools and modes – but our experience with amateur users causes us to focus on additional aspects: how can the tools usefully and productively embody appropriate assumptions about what users are trying to do, in the way that consumer products designed for amateurs typically do? And how can tools be made easy for users without having an overly formative effect on the sorts of things users then make?

**The Novice and the Amateur**

What do creative amateurs need? We begin by looking at the more familiar concept of the novice. There is an extensive literature on the special problems of designing for novice users, and a range of proposed solutions, often found in combination: wholly external support such as improved training [1]; semi-external support such as online help (for example [13], [8], [36]); more integrated assistance such as the generic Microsoft ToolTips and Macintosh Balloon Help; and task-specific integral help (eg. [20]). However, the discussion of most interest here is that on truly internal support for novices, that is, where the system is itself designed or redesigned with the needs of novice users in mind. Given the widespread propagation of computing from specialist professionals to ‘ordinary’ users, it might be assumed that nowadays simply to design a system well is to make it accessible to novices (or to put it inversely, that to make it accessible to novices is to ensure that it is well designed), and no doubt there is some truth in this. In some cases it seems best to treat all users as novices, for example in designing a kiosk interface for purchasers of train tickets [35], where it hardly makes sense to cater specially for the needs of an expert user. But there is an ongoing discussion of what may distinguish systems designed for novice and experienced users and particularly how their requirements might conflict (for example [12], [7], [30]). Sometimes the problem is approached by suppressing complexity at the initial encounter but making it more accessible as the user progresses [9], [15].

Here we develop the discussion by introducing the concept of the amateur. Such a user is different from a novice in the single important respect that he or she is not assumed to be an expert-in-the-making but is expected to remain a quite unsophisticated user. This user will, we argue, continue to require an interface different from that of the
expert. An additional factor, as Bederson [3] points out in discussing the design of image browsers, is that even with nominally similar requirements, amateur users may actually be doing subtly different tasks as well as needing different means to do them.

FIVE TRANSFORMATIONS
What is it that distinguishes systems for amateur use? We discuss five transformations which enable systems previously only accessible to experts to be usable by the amateur. We are not suggesting that any one transformation is unique to design for amateurs – rather that the combination, and the intensity of application, is special to this discipline. The five transformations have some overlap with each other. Rather than concentrate solely on digital interactive systems, we will also consider examples from product design (such as the camera, which we claim offers a stimulating exemplar).

Transformation 1: Foregrounding
In the first place, while a camera for experts may at first present a largely inexplicable and complex set of interfaces and controls, one designed for an amateur is expected to make its most important functions instantly accessible. The user takes the camera from its case and instantly guesses which button is used to take the picture (assisted of course by cultural knowledge). Such readability in designs has been extensively studied by Norman [22] and within the field of Product Semantics (eg. [16]). Even with systems which are less streamlined such as (notoriously) video recorders, at least the documentation is usually presented to emphasise instant access to the primary functions: ‘Playing a tape straight away!’ ‘Making a recording straight away!’ and so forth. We can call this prioritisation in the design ‘foregrounding’. Often it is manifested to the user as ease of getting started, or a feeling of directness – that nothing comes between the user and their purpose – the perfect Heideggerian tool which is transparent to the task or ‘ready-to-hand’ [38]. To achieve this the design is based on assumptions about what the user will most likely want to do. But surely all design is based on this – the essence of requirements capture? In design for amateurs, however, the processes of foregrounding and backgrounding (discussed next) take much further the business of assuming what users will want to do. This is a risky strategy: if the decisions are the wrong ones then the system will be difficult and unsatisfying to use. But if the decisions are the right ones, the user will experience the system as liberating rather than constraining. Later we contrast the assumptions which have been made in designing software – particularly, in our example, for avatar creation – with the assumptions which we believe should have been made.

Transformation 2: Backgrounding
As some functions are brought forward others are suppressed. This is perhaps the aspect most likely to offend the expert user. It involves partially concealing certain functions, or even making them completely unavailable, for which we borrow the term ‘backgrounding’. Returning to the example of the amateur’s camera, it is normal for the design to assume that users do not want to use flash lighting when there is adequate ambient light, but that they will want to when there is not. If users wish to override these settings, either they cannot, or they must engage in a relatively complex and obscure set of button presses to achieve it. Clearly a price has been paid for the process of prioritisation. This difference in design for the amateur and the expert is in some ways analogous to Norman’s distinction between wide-deep and narrow-shallow systems in his consideration of human error (an issue clearly relevant to designing for ‘fool-proof’ use) where he argues that deep and wide systems – in his example, strategy games – are designed for conscious thought, while narrow and shallow systems – such as speed and reflex games – offer scope instead for subconscious skills [22:119-123]. Applied to the amateur-expert dichotomy (if such it is), we might say that deep-wide systems are best suited to experts, narrow-shallow ones to amateurs.

In computer systems, a clear example of foregrounding – and its correlate backgrounding – is the graphical user interface. While the command line interface (CLI) makes all of its commands almost equally accessible (or inaccessible), the GUI is fundamentally layered. Using a command line, users may operate on objects they cannot already see represented on the screen (for example by a filename) as easily as on those they can see. In a GUI, users are assumed to want to operate only on those things which are visible (Boyd Davis [6] discusses the unwanted formative effects this has on the design of many kinds of interactive media). This also helps to prevent them from doing operations which are ‘dangerous’, on the grounds that operating on things you cannot see is a risky business. The popular debate over the merits of the GUI and the CLI continues to rage, but the often unspoken consensus seems to be that the CLI suits many expert-level tasks while the GUI suits the needs of the amateur. Thimbleby et al. [35] in their critique of a ticket-purchase system also advocate concealing the complexities of underlying systems and in so doing oppose Norman’s view that the design should allow the user’s conceptual model to approach as far as possible the system model [21]. Again we could regard the former approach as prioritising the needs of the amateur while the latter is more appropriate to the expert.

Transformation 3: Automation
Cameras for amateur use generally offer an auto-focus facility. The exact design of this automation tells us a lot about designing for the amateur. Based on the knowledge that users customarily position the primary subject of their photograph in the centre of the frame, this is the area prioritised by the system for focussing. Usually the user can ‘trick’ the device into behaving differently, but this is essentially a piece of expert, non-amateur control to override the knowledge built into the device. Similarly, if an automatic camera detects that the central region of the
image is darker than the periphery, it ‘assumes’ that flashlight is needed – because amateur photographers often accidentally silhouette their subjects. While to the expert such a camera would be generally intolerable, for the amateur the machine-embedded knowledge about photographs yet to be taken is of great utility in ensuring successful results. The amateur’s camera is thus a valuable ‘cognitive artifact’ in Norman’s sense [23]. In Sontag’s disparaging terms, ‘taking pictures [with an amateur’s camera] demands no skill or expert knowledge ... the machine is all-knowing’ [33]. It should be noted that we are not talking here about what is commonly called an ‘intelligent interface’: the camera is simply engineered around assumptions based on human best practice.

Clearly the ideas of automation, of foregrounding and backgrounding are interrelated, all aiming to provide amateur users with what they are most likely to need. Importantly, the popularity of amateur products in a competitive marketplace suggests that the decisions taken are generally the right ones.

Transformation 4: Constraining
One of the most valuable things the designer can do is to protect amateur users from their own mistakes, and here the parallels between product design and software design are most obvious. The concept of interlock, which strictly enforces certain user behaviours [22:135-7] is now highly familiar and indeed is commonplace engineering practice: the boiler cannot be opened until the pressure-release valve is released; the meat-mincer cannot be operated without closing the lid. The GUI prevents the user from closing a file without negotiating the Save dialog while the photographer cannot use the shutter until the film has been advanced. Of more interest here is the use of less rigid but nevertheless effective constraints or forcing functions which favour certain user behaviours. Physical examples include the trainer wheels on a child’s bicycle which protect the novice but would certainly constrain the expert activities of an experienced stunt-biker, and camera lockouts which oblige users to expressly confirm their decision to take a photo in inadequate light by making them manually override the system defaults. Such forcing functions are clearly close relatives of the foregrounding, backgrounding, and automation already discussed, sharing with them the fundamental idea that ‘expertise’ has been embodied in the artefact based on assumptions about proper use.

Transformation 5: Integration
Photography has been greatly simplified by the Polaroid camera and now by digital cameras where users can see the output instantly. WYSIWYG displays did the same for writing and publishing. Early text-only coding tools for creating HTML pages have been widely replaced by graphical layout tools (though, significantly, many professional Web developers still prefer the control that hand-coding provides). In all cases the appeal lies in the tight integration of action and feedback, a familiar principle of direct manipulation [31:202-204]. But this assistance is of most value to the amateur, because the expert usually has enough ‘knowledge in the head’ [22:54-80] to work without this externalisation. The experienced composer can write a symphony using paper notation, where the amateur must hear actual sounds. The lack of integration of tools and of modes within tools, which would have enabled in situ feedback, was certainly one of the primary impediments to creative activity in the VERTEX project, which we now describe.

SCENARIOS FROM THE VERTEX PROJECT
In this section we discuss the applicability of the five transformations outlined above to the issues raised in a practical creative project. Based in three UK primary schools, VERTEX has involved over a hundred children and their teachers. These schools are in very different geographic and cultural locations – at the heart of London’s West End, in suburban outer London, and in a small coastal town on Orkney, off the north coast of Scotland. Each school has a diverse social and cultural mix, including children learning English as a second language and children with a variety of special educational needs. Both the teachers and children have varying levels of access, skills and experience in uses of new technologies.

Project activity was centred on the virtual world software ActiveWorlds V.2.2 (ActiveWorlds Inc.). It also incorporated both digital and non-digital approaches to developing ideas, progressing designs and making 3D objects. Children used new media in combination with traditional activities, ranging from story writing, collage and drawing to the use of software tools including Photoshop (Adobe Systems Inc.) and 3D Studio Max (Discreet). In each school they worked together as a class, in small groups, and through online teaching sessions, imagining, planning, designing, and making their own virtual worlds and a unique set of avatar characters. A typical creative cycle might involve in outline the following activities: 1. sketch, paint, and collage a design for an avatar, object or scene; 2. scan artwork and take digital photographs to create texture-maps; 3. build a simple digital model and apply texture-maps; 4. articulate avatar or modify model; 5. upload avatar into shared 3D space (these last two were completed by staff). Other activities such as online drama, storytelling and exhibitions took place within the built worlds.

The young learners had a number of difficulties, but it is important to note that these did not generally arise from conceptual problems with understanding what the tools did, nor with working in a virtual three-dimensional space. On one occasion we witnessed a child considered by her teachers as of average ability giving a fluent and fully competent account to another child of manipulating objects in (x, y, z) coordinate space. In that sense there was nothing ‘childish’ about their problems.
The children demonstrated a remarkable facility for designing and building their avatars and objects, even when using professional authoring tools such as 3D Studio Max and Adobe Photoshop. Using these tools clearly has learning benefits in terms of general and specific Information and Communication Technologies (ICT) skills that will enable the children to produce work beyond the lifetime and scope of the VERTEX project. However, the research identified several problems associated with amateur use of such software (we answer below the obvious question why we used professional software with young children). The various problems in using packages aimed at the amateur are in themselves instructive when considered in terms of the five transformations.

**Scenario One: Building Models**

At the outset it must be said that the authoring mode in ActiveWorlds is strikingly poor in terms of usability by almost any standards. Nevertheless, after an evaluation of the small range of competing products, we chose this system because even with these problems it was on balance the best available at the time (there have been some recent welcome developments which we discuss below). In this scenario we outline the sequence of actions which must be undertaken if the user wishes to insert a library object into the shared space. The user 1. consults a help system which gives a list of filenames of standard objects in a server-based library; 2. remembers or writes down the name of the desired object; 3. goes to the scene; 4. selects an existing object (any object) in the world; 5. right-clicks to open a dialog allowing duplication of the object; 6. enters the model-name previously memorised or noted; 7. closes the dialog; the new object appears. This indicates how far some desktop virtual world construction tools have yet to go to achieve basic standards of usability. In relation to our five transformations, a number of things are clear: the actions which the user is most likely to want to undertake have not been foregrounded; obscure system-level operations have not been pushed to the background as they should; users are expected to do for themselves things which should have been automated; and this is a batch-wise operation, which does not allow in situ feedback through integration of the user’s work into the shared space until after an extended process has been completed. Almost any form of heuristic usability analysis would have yielded a useful critique in this extreme case: we do not claim that our five transformations have a special validity here, but note that the problems are well captured in our terms. Other scenarios allow us to emphasise the usefulness of our approach.

**Scenario Two: Creating a Bespoke Object**

Many of the difficulties encountered have their root cause in basic weaknesses of integration. Several potential choices of tools were unavailable to us because of incompatibilities of file formats between modellers and desktop virtual worlds software. There are signs that these problems are diminishing in newer tools, but there is still a tendency for modelling tools aimed at the amateur to be based on unwarranted assumptions about what users will want to do with the finished model, in terms of the file-export formats available. Such problems are familiar from most computing domains but are acute in this area. A particular problem arises when a file format is nominally ‘supported’ but in practice many of the attributes in the file are ignored.

As already indicated, the making of bespoke objects was important for this project: we encouraged the users to make objects which were personally meaningful and which allowed the embedding of the digital processes in a cycle of familiar physical designing and making. In ActiveWorlds, the cycle of digital operations to insert a bespoke object in the world involves making the object in a third-party 3D modeller, FTP-ing it to server (having noted its filename because this does not then appear in the library), and then using the process already outlined for standard objects: entering the scene; selecting an existing object; duplicating the object; entering the model-name; closing the dialog to cause the object to appear. In this respect, there are now signs of greater maturity in what is available: in Atmosphere (beta release, Adobe Systems Inc.) users can build in the world using simple primitives (bespoke objects are imported from local files). Integration of the create and
view modes means users can view what they are making \textit{in situ}. They subsequently share what they have built. However this unfortunately rules out something which is possible in ActiveWorlds, namely shared building, where users collaborate in the shared space to assemble an object together.

For the external modelling, the users built using 3D Studio Max Versions 3 and 4 (Discreet), a professional modelling package, partly because, as indicated, none of the more amateur-oriented software supported the file formats we needed to use. 3D Studio Max offers an interface designed for professionals which not surprisingly is somewhat complex and intimidating: it is both wide and deep. However, it is clear that even this professional application makes use of foregrounding and backgrounding, for example in its hierarchy of ‘roll-outs’ which allow progressively more attributes of objects to be user-specified if the user wishes to do so. Embedded assumptions about how users will want to proceed – embodying knowledge in the device – have been refined over several iterations of the package.

More questionable are its other assumptions. A hierarchy of panes of objects assumes priorities not just about \textit{how} users will want to proceed but about \textit{what} they want to make. Priority is given to regular, geometric, inorganic shapes, strongly determining the kinds of things that are made, especially by amateurs who, of course, tend to operate with the layer of the interface which has been most strongly foregrounded. To return to the analogy of the camera, it is as though the foregrounding were to determine what subjects the user made pictures of, rather than simply ease their task in taking satisfactory pictures with the minimum of risk and difficulty. Moving on to consider modelling packages more strongly oriented to amateurs, this tendency becomes still more pronounced. Such channelling of users’ activities almost approaches ‘painting by numbers’ as compared with real painting, the former clearly in our definition not a creative activity.

In 3D Studio Max all the standard geometric modifications are possible. As part of the embedding of the digital work in non-digital design activities, an important part of the project was for our users to apply real textures collected ‘from the wild’ using digital cameras and other techniques. Unfortunately loading and applying textures in 3D Studio Max is a six- or seven-part process. Such lack of integration caused by excessive modality is not just a simple usability barrier in terms of efficiency; it is positively counterproductive to creative experiment. The integrative transformation required here is one which would enable users to achieve the rapid and fluid externalisation, experimentation and reflection called for by Shneiderman.

When it comes to inserting the model made with a third-party application into the virtual world, further barriers to creative work are presented. Our young amateurs successfully managed to repeatedly overcome these barriers, but the tools created an onerous training need – there was no possibility of learners safely finding their own way as they might with a suitably designed tool; they tended to lose focus on the task, despite high levels of motivation; and there was again a painful lack of flow, caused by the severe lack of integration. For instance on one occasion, a user created a sofa in a third party application which, on loading into the virtual environment, turned out to be so colossal as to obliterate everything else in the world. In itself this would not matter, but there was no remedy within the virtual world: the object had to be scaled in the originating package and FTP-ed again to the shared space. In being forced to undertake this additional loop the user was disproportionately ‘punished’ for a single mistake which a series of properly integrated constraints could easily have prevented.

\textbf{Scenario Three: Creating an Avatar}

The process of making a personal avatar was an important part of the creative and educational strategy for this project, enabling the learners to become strongly engaged in the worlds they were building. Much of their creative progress – and satisfaction – arose from collaborative meetings with their fellow learners in the space and from the processes of externalisation and self-representation. Once again, many opportunities arise for transforming existing tools into ones more suited to amateurs.

The users made models in 3D Studio Max based on their own simple articulated puppet designs which they had constructed in cardboard, paint, cloth and other physical materials. The package, using the CharacterStudio add-on (also by Discreet), makes easy the construction of bipedal humanoid figures, but many of the personal avatars did not fit this preconception. One child, asked why his puppet had only one leg, said that he had realised a one-legged creature could stand up just as well in a virtual world as a two-legged one. The ‘foot’ of one puppet was a skateboard. Many other divergences from ‘normal’ anatomy were also wanted. The package on the other hand so far presupposes what users will want to make (again instead of making easier the building of whatever \textit{users} choose to make) that a foregrounded option in the humanoid modelling is a ‘ponytail’.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{avatar_images}
\caption{Physical ‘puppets’ for translation into avatars in the virtual world.}
\end{figure}
The next step was to articulate the model, specifying the joints between sub-model parts and the kinematics hierarchy. Is should be noted that none of this was conceptually problematic for the young users. However the software routines for this operation take no useful advantage of assumptions about what is likely to be needed. For example, if two sub-objects overlap or adjoin, the software does not make the reasonable assumption that a joint will be required there. And uncalled for precision is required: instead of the software offering likely default positions for joints, the user must position each one ‘by hand’ in (x, y, z) space. Much could be learned here from the free-form table-drawing of Microsoft Word, with its ability to resolve rather approximate mark-making into a structured and indeed hierarchical table. The utility package Accutrans (MicroMouse Productions) uses naming conventions for sub-parts (‘lu_arm’ for left upper arm, for example) to help create model hierarchies, but this simply locks users again into conventional presuppositions about what they want to make.

As befits a professional package, in 3D Studio Max users can do almost anything they like if they have substantial expertise. Using the CharacterStudio add-on many things have been automated, such as model hierarchies, but here again the foregrounding is based on an assumption of humanoid bipeds. The supplied locomotion files also assume conventional forms. Poser (Curious Labs Inc.) is more accessible — and AvatarLab is a trimmed down version of Poser that can export directly to Adobe Atmosphere — but both foreground bipedality. There are no models of children in the supplied library and it is clear that the packages are based on assumptions of making certain kinds of figure. A range of parameterised models is provided, but again the kinds of changes which can be made are extremely limited. The assumptions made are not, as we suggest they should be, about processes: they are about things.

CONCLUSIONS
Children were trained in the use of 3D Max, Photoshop and other ancillary software. We were obliged to use professional tools because only these gave us the openness and level of control we required together with the file-formats we needed. The means by which the training was carried out were refined over the course of the research to reduce skill acquisition time and also to provide an appropriate level of skills to suit the necessary tasks. Nevertheless, the complexity of these professional tools means that this process still takes longer than is desirable and is beyond what would be possible outside a research project. The time cost for becoming competent in using professional tools is unrealistic. Professional packages are also expensive and therefore not a practical proposition for the creative amateur (whether in or out of school). What we needed was software that would support amateur users through the kinds of processes we have outlined, without excessively constraining what they could make.

The poor choice of software options available trends to mean that either the experience threshold at which amateur users can achieve tasks effectively is too high, or the opportunities in terms of what they can make are too constrained.

Delays – and, more importantly, breakdown of the creative cycle – resulted from such things as manual file translation and transfer. These require automation, or even elimination by removing excess modality. Many impediments arose because of the separation of 3D modelling and world construction. The virtual worlds browser allowed the children to make environments and structures, but not to make the constituent models themselves. This separation of model making and world building leads to a lack of immediacy, loss of engagement and task focus in the overall process.

Most virtual worlds software assumes that amateurs are just consumers, whose role is to browse and explore worlds, and that the building of worlds will be confined to experts. A clearer grasp of the special characteristics of designing for amateur users – creators who are also consumers – would lead to improvements.

This research suggests (1) an immediate approach and (2) a longer term strategy to overcome some of these problems:

• There is a need for 3D world browsers with integrated modelling tools designed with sufficient functionality and simplicity of interface to enable competent 3D world building. Such packages are starting to emerge. We are ourselves exploring the potential of creating a tool in Macromedia’s Director 8.5/MX or Java, as well as evaluating Adobe Atmosphere. However, it is as yet unclear whether readymade tools will strike the right balance between offering a wide range of creative functionality which enables consumer-creators to make what they want to make – essentially giving them the creative freedom that an expert enjoys from professional software – and at the same time supporting the user through the five transformations from design for professionals to design for amateurs that we have described.

• The longer term strategy should be to develop and refine the approach proposed here, which focuses on the specific needs of amateur users. A clear view is required by the HCI community of the means by which best practice is, and can be, transferred from the use of tools by experts into the engineering of software and other products for non-experts. The best product designs already provide instructive exemplars of this approach, which has enabled technologies previously accessible to the few to be transformed into satisfying and enjoyable tools for all.

ACKNOWLEDGMENTS
The research which led to this paper was funded by Middlesex University through an allocation of its HEFCE
Non-Formula Funding for Research. The authors gratefully acknowledge this invaluable support.

REFERENCES
24. Ong W.J., Orality and literacy – the technologizing of the word, Methuen, London, 1982


