MICHAEL'S COMPUTER GAME: A CASE OF *OPEN* MODELLING

Ronnie Goldstein (Open University, UK) & Dave Pratt (University of Warwick, UK)

We report on work from the Playground project, in which young children (6 to 8) are designing their own computer games using a specially built iconic language to represent rules that determine the behaviour of objects in the emerging game. We report on one child's development of a maze game to illustrate how conventional modelling theory might be extended to encompass situations in which children legitimately construct and reconstruct the problem being modelled and in which children are simultaneously learning about the potential and utilities of the modelling tools themselves. We conclude that a framework for such modelling needs to acknowledge the complexity of activity and to incorporate new perspectives on validation and on modifying the goals.

Introduction

The notion of modelling that interests us describes the activity of creating and testing a model that represents critical elements, including typically the mathematical structure of a problem or system. Much of the research on modelling activity involves the use of software; indeed Schecker (1993) proposes that model building software be recognized as a category of tools in its own right, and defines this new category as:

> "… context-free software tools that support the user in representing a part of the 'touch-and-show' reality in the form of an abstract, quantifiable system of parameters and their relationships (the *model*), which predicts the behavior of the real system." (p. 162)

Such software has been used by students to explore ready built models (for one of many examples, see White, 1984) and to express the learners' own ideas for models that represent a particular situation or problem. (this distinction is discussed by, amongst others, Bliss & Ogborn, 1989.)

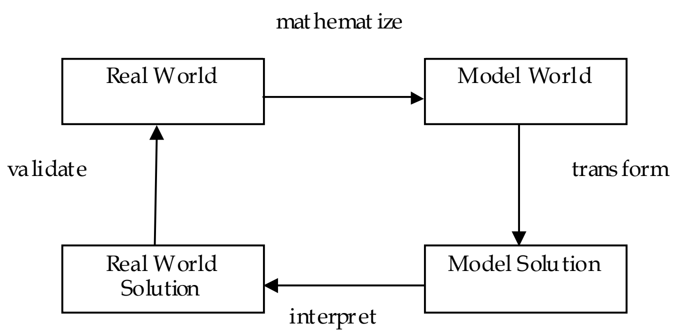

Fig. 1: The modelling cycle

Our interest in modelling stems from our work in the Playground Project[i]: we are studying young children (6 to 8 years old), as they use specially designed software to make their own computer games. Our focus is on the way young children express their ideas for games through the articulation of rules. We wish to modify or extend modelling theory to encompass situations like ours where the learner possesses considerable control over the nature of the problem: the child is involved in problem posing, an activity that falls outside the classical modelling cycle as expressed in Figure 1.

We also hope to adapt the modelling cycle by acknowledging the central role of tools in shaping the activity. Our starting point owes much to the situated cognitionists (Lave, 1988) in that we regard tools and learners to be in a dialectical relationship; that is to say, as the learners become more familiar with the tools, they become aware of new opportunities and utilities of those tools. Through using the tools, the learners re-construct their understanding of them. This shapes the way that the learners think about their solution to the problem, and, where problem posing is legitimate, the problem itself.

In summary, we aim to illustrate the evolutionary nature of a form of modelling that encompasses conventional modelling activity but adds to it two important processes: (i) an openness that makes it legitimate to change the problem being modelled, and (ii) an interleaving of learning about the tools themselves.

Our approach

We report the work of one child, Michael, who is 7 years old. He has been building a maze game using *Pathways*, one of the pieces of software being developed in the Playgrounds project. *Pathways* is driven entirely through the use of icons with the intention that it is accessible to very young children. The software allows children to choose from many different background screens and to place a number of objects on the screen. The objects can be given various shapes and they can be made to move automatically or to be dependent on the joystick or the mouse. A significant feature of the software is that objects can be given rules that determine their behaviour when the game is played. An underpinning hypothesis is that young children can create complex behaviour by teaching objects a relatively small number of simple rules expressed in an iconic language. An object can hold several rules. For example, the pathways in Figure 7 apply to the walls of a maze and they express the rules: ‘When I touch the tiger, I reveal a rabbit’ and ‘When I touch anything, bounce off me’ (i.e. any other touching object should bounce away from the object that holds the rule). Children move from defining rules to playing the game by clicking a switch that turns the game on but they can return at any time to making rules by switching the game off again.

The game evolutions are captured in clinical interviews, in which we work as participant observers alongside pairs of children (usually pairs, though in the episode reported in this paper Michael is working on his own). The children are between 6 and 8 years of age. The interviews normally last between 1 and 1.5 hours. Game evolutions usually extend across 4 or 5 such interviews.

The data for the research are transcriptions of the recordings on videotape of the children’s activity, captured from their computer screen and from discussions between the children and the researchers. Also, there are many versions of the emerging game, saved at regular or significant moments.

Evolution of a Maze Game

This episode began towards the end of the second session with Michael, who was building a type of maze game in which the player controls the tiger with a

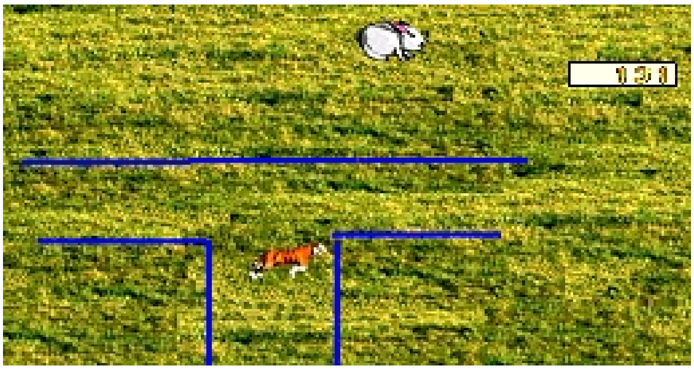

Fig. 2: Michael’s game at the beginning of this episode

joystick and tries to 'catch' the rabbit (see Figure 2) at which point the rabbit disappears as if eaten by the tiger. He has built a timer (in the top right corner of the screen), which counts the time that has elapsed since the game was switched on. The game contains several pieces of wall (the blue strips) but so far Michael has not given the walls any rules. This episode describes how Michael's game evolved to include rules for the walls. (Note: In the transcribed sections below, M refers to Michael and R to the researcher(s) who were present.)

At the beginning of this episode, Michael played the game and found that the tiger passed straight through the walls. Michael wanted to turn the walls into barriers.

1. M: Can we make it not be able to go through?
2. R: There is something we can try on that but let's worry about that a bit later.
3. M: Would it be the bounce button?
4. R: Yes. How did you know about the bounce button?
5. M: When I was searching.

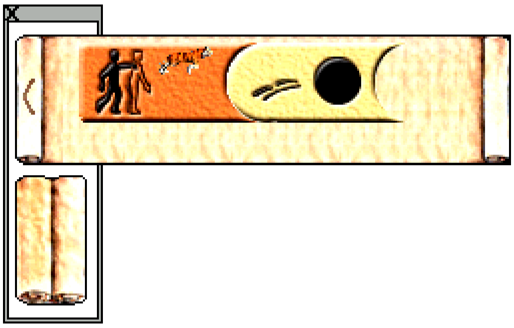

Fig. 3: Michael's rule for bouncing

Michael had remembered some earlier exploration in which he had come across an icon for bouncing. When this icon is used in a rule, anything else that touches the object with the rule bounces off that object. Michael opened up the pathways for one piece of wall and began to enter each element of the wall's rule: 'When I am touching the tiger, bounce off me' (Figure 3).

Michael checked this rule by playing the game but found that the tiger passed straight through the wall. The researchers explained that there was a conflict in Michael's rules: Michael had already written a rule for the tiger which meant that it was controlled by the joystick and so the tiger did not know whether to follow the joystick or to bounce off the wall. In fact, the tiger would momentarily bounce off the wall but then immediately pick up the position of the joystick again. Michael was not content.

6. M: I wish it would just not go through … it wouldn't really be a proper maze because the idea is that you can't go through stuff like walls and stuff. That's the whole point of mazes.

With some encouragement from the researcher, Michael began to consider alternative solutions that would resolve the bouncing conundrum. He considered penalising the player somehow when the tiger collided with a wall.

7. M: If you touched the wall, I was thinking …Oh, I've actually just had an idea. It changes its speed. It speed goes up so it's harder to control. So basically it goes, "Oh no I've touched the wall". It gets faster and faster and faster.

The researcher explained that the speed of the tiger was controlled manually by the joystick, and so any programmed change to the speed of the tiger would in any case be overridden by the player.

8. M: I've just thought of the worst one ever. The game stops.

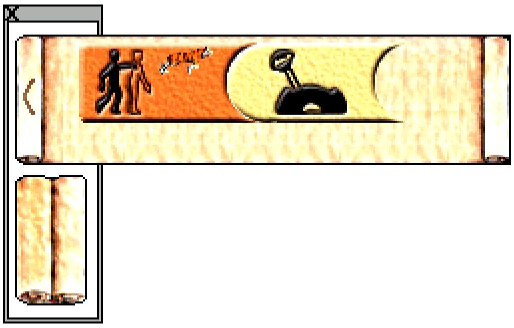

Fig. 4: Michael's new rules for each circular obstacle

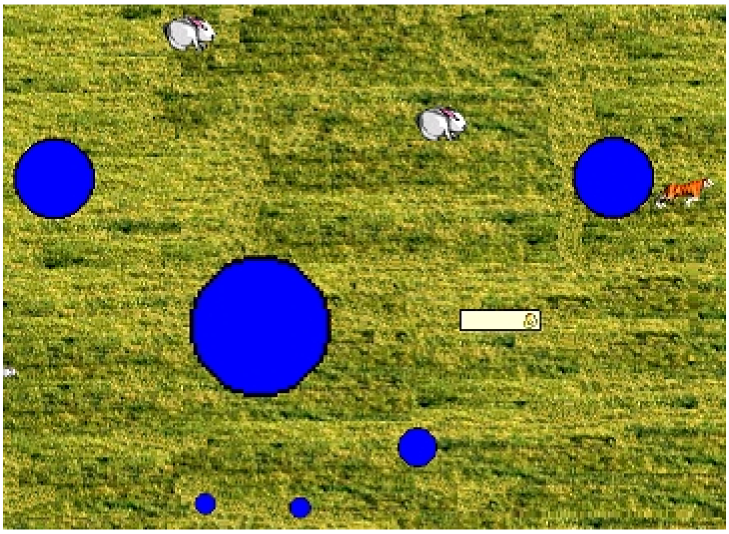

Fig. 5: Michael's game with circular obstacles

Michael entered the rule: 'When I touch the tiger, I stop the game' into a blank pathway for one piece of wall (Figure 4). He then copied this rule for each of the other pieces of wall.

Michael kept this rule for the walls for most of the remainder of his time working on the maze game. During this time he concentrated on other aspects of his game. By the time we pick up the story once more, Michael has changed the walls into circular obstacles (Figure 5). He has also added two extra rabbits, each of which moves automatically and bounces off the walls.

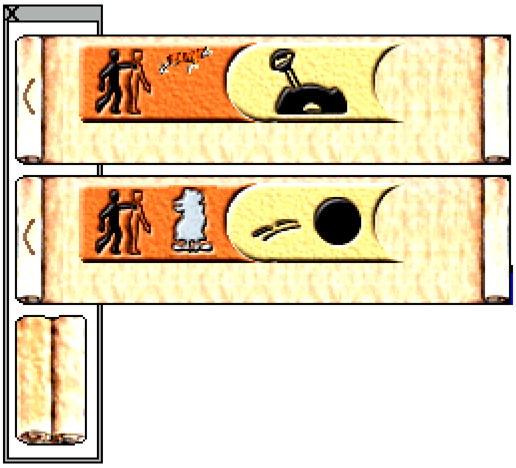

Fig. 6: Michael's stopping rule for the walls

Michael had created this effect by applying the earlier idea of bouncing off the walls to the rabbits. Thus, each circular obstacle contained the pathways in Figure 6.

The second rule can be read: 'When I touch anything, bounce off me'. Michael had already learnt that the tiger was controlled by the joystick and could not (visibly) be made to bounce off the walls, so in effect only the rabbits would now bounce off the obstacles.

However, Michael was still worried about his first rule, which made the game stop too often. He wanted to penalise the player some other way when the tiger collided with an obstacle.

9. M: Can I make … oh yeah … I could change that rule (pointing to a circle) until …make it, could I make it one rabbit appears every time you touch it?

The researcher suggested that Michael could start with some hidden rabbits and make them appear when the tiger touched an obstacle.

10. M: …also when you … like you can destroy one, you can get one but then you go in … and you catch one, it appears again but, would that happen?

The researchers asked for clarification.

11. M: Well, tiger gets the rabbit but then he accidentally goes into a ball and the rabbit appears again, sort of.
12. R: Oh I see, one that's been eaten appears.
13. M: Yeah.

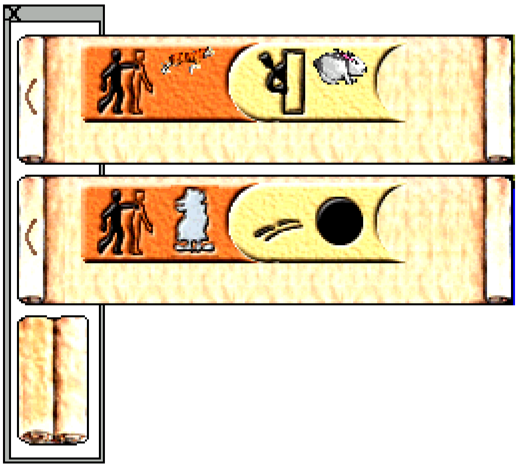

Fig. 7: Michael's rule for making the rabbits re-appear

Michael decided that since he has six obstacles and three rabbits, he could allocate two obstacles to each rabbit, so that whenever the tiger touched an obstacle it would make a particular rabbit appear and, if that rabbit was not already hidden, nothing would apparently happen.

Michael erased the 'I stop the game' icon from the pathway of one of the obstacles. With some help, he found an icon for 'I show' and inserted it into the rule (see Figure 7) and he made sure that the correct rabbit was included in the rule. Michael repeated this process for each obstacle.

Discussion

We have noticed three very different phases of activity in the pupils' work. These phases have no clear order and they are often not distinct from one another.

Phases of activity

1. Game-oriented activity (G): There are some points during the sessions when the main focus of the students' work is on the game itself. When the pupils are in G their thinking is focused on the final outcome of their efforts, i.e. the game they are designing. They are concerned with issues such as the context for the game, the background screen, what happens, who controls which objects, how many players are involved and how any victory is accomplished. Lines 6, 7 and 8 above are comments from Michael at times when his thinking was clearly game-oriented.

2. Tool-oriented activity (T): There are other stages during the pupils' activity when they need to learn about the tools that they are using. Whenever computers are used, some time needs to be devoted to understanding the software and how it operates. This is quite normal and there are bound to be stages when the pupils have to find out how to change the background picture on the screen or which particular icons in which order might be used for one of their rules. In line 5 Michael refers to a tool-oriented stage when he was "searching", by which he means he was exploring the software.

3. Rule-oriented activity (R): The pupils create particular rules for particular objects in their game to make things actually happen when the game is played. Michael has given a rule to the walls in his game so that when the tiger touches any of them it will bounce, rather than carry on in the same direction (Figure 2). Later he changed the rules for the walls (which had become circles) so that when the tiger touched one of them a new rabbit was made to appear (Figure 6). When the thinking is clearly focused on a rule to be included in the game, we say it is rule-oriented activity.

Complexity

Michael's activity highlights two issues that have implications for how we think about the modelling cycle.

1. Order of the phases: It must already be clear that there is no prescribed order for the three phases (G, R and T) described above. For instance, the students are engaged in game-oriented activity when they use the icon in the toolbar at the foot of the screen to switch their game on. This might happen at any stage, either for the sheer fun of it or to validate their changes. Thus the place of G in the complete picture cannot be determined in advance. It is also evident that T and G might occur in either order: on occasions students learn about a particular tool before they attempt to use it in the game (line 5); at other times they learn about a new tool because its need has become evident in the game (activity leading up to the rule in Figure 7). The preferred approach will vary with the student and with the particular situation.
2. Links between the phases: The complexity of the phases is also attributable to the close links that exist between them. Activity in G often suggests new possibilities for R (see the text leading up to and including line 1). In lines 6, 7 and 8 Michael is in G but he is moving towards R because he is starting to think about particular rules. During the course of this short extract he uses the phrase "If you touched the wall" which is very similar to the one used by the software's mouth when it is applied to certain rules. Students need to learn to think about their game's rules formally in the language of the software so they can change the game and Michael has started to do this very early. There is, however, no clear link in the data that has been collected between T and G and this leads to the hypothesis that R is critical within Pathways and an equivalent mode needs to be found for other situations that depend on the acquisition of tools.

Modelling

Our analysis of Michael's work highlights several features that are similar to those in the conventional modelling cycle. We see analogies between the real and model worlds of the conventional modelling cycle with Michael's game-oriented and rule oriented activity. We also recognise the processes of interpretation and validation. Furthermore, in describing the complexity of the interleaving of the different phases of activity, we are observing no more than has been reported by several other researchers who have commented on the lack of the linearity implied by the conventional modelling cycle (Lesh & Doerr, 2000). Because of the commonality between Michael's activity and modelling theory, we believe it is productive to consider the episode as a case of modelling. Nevertheless in the next two sections we wish to elaborate on some aspects of the data, which strike us as different from conventional modelling.

De-goaling

Michael resolved the conflict over bouncing by changing the game so that it simply stopped whenever the tiger touched a wall (line 8). Michael forged new insights into the utility of the tools involved in trying to make the tiger bounce and this caused him to redefine his objective. Later he adopted the bouncing idea for the rabbits (lines 9-13) and so changed the aim of his game once more. A feature of the above episode is that, during his tool-oriented activity, Michael often sought out new possibilities, and exploited new discoveries. In an earlier incident, Michael explicitly recalled his prior experimentation (lines 2-5) and subsequently used the bounce stone during a phase of rule-oriented activity in order to create the effect of walls as boundaries (Figures 3 and 4).

We refer to the process of changing the aim of the game as *de-goaling*. The purpose of the activity, as understood by Michael, was to create a computer game. This explicit purpose made legitimate, indeed encouraged, de-goaling, a process that is not normally associated with modelling.

Validating

Another feature of Michael's activity is the way in which he validated his ideas. The move from R to G was often triggered by the need to validate his recently expressed rules. In the rule-oriented activity that preceded line 6, Michael had formulated a bouncing rule (Figure 3). He validated this rule by playing the game. On this occasion, the validation was initially negative – the rule did not work – but subsequently very positive since he found new ways of resolving the dilemma as discussed above.

Similarly, the creative game-oriented activity in which he imagined disappearing and re-appearing rabbits (lines 9-13) is stimulated by the validation process (playing the game) that triggered a dissatisfaction with the game stopping repeatedly.

We are struck by the natural way that Michael moved from the model-world of rules to the 'real' world of his game because of the constant need to validate his progress and the legitimacy of changing the problem being posed, his maze.

Open Modelling

The feature of de-goaling and the nature of validation lead us to believe that we should regard Michael's activity as a variation of conventional modelling. We refer to the type of modelling illustrated by Michael's activity as *open* modelling. Open modelling describes activity where the learner is encouraged to reformulate creatively the problem itself as a result of learning about the utilities of and potentials for the modelling tools. In the literature on conventional modelling, concerns have been expressed about the lack of evidence of validation strategies when children are expressing their own models (Doerr, 1996). Matos (1995) reports on 10^{th} grade students using spreadsheets to model the length of paper in a roll of given diameter. Matos found that the spreadsheet's use was transformed during the activity: the software started as a tool for expressing models but became the reality within which the students worked. As a consequence, Matos' students hardly ever referred back to the problem after the initial stage. Validation rarely occurred in the conventional modelling cycle. In contrast, Michael engaged naturally and almost continually in validation processes. The creative challenge of de-goaling provides a drive to move between real and model world activity. There is some evidence that, because of the legitimacy of changing the problem being modelled (i.e. de-goaling), validation occurs more naturally and takes on a more positive and creative role in open modelling than it does in conventional modelling activity reported by other researchers.

References

Bliss, J., & Ogborn, J. (1989). Tools for Exploratory Learning. *Journal of Computer Assisted Learning, 5*, 37-50.

Doerr, H. (1996). Stella Ten Years Later: A Review of the Literature. *International Journal of Computers for Mathematical Learning, 1*(2), 201-224.

Hoyles, C., Noss, R., Adamson, R., & Lowe, S. (submitted). *Programming rules: What do children understand?* Paper presented at the Proceedings of the Twenty Fifth Annual Conference of the International Group for the Psychology of Mathematics, Utrecht, The Netherlands.

Lave, J. (1988). *Cognition in Practice*. Cambridge: Cambridge University Press.

Lesh, R., & Doerr, H. (2000). Symbolizing, communicating, and mathematizing: Key components of models and modeling. In P. Cobb & E. Yackel & K. McClain (Eds.), *Symbolizing and Communicating in Mathematics Classrooms : Perspectives on Discourse Tools and Instructional Design* (pp. 361-383). Mahwah, NJ:: Lawrence Erlbaum Associates.

Matos, J. F. (1995). The Spreadsheet as a Tool for Mathematical Modelling: A Case Study. In A. A. diSessa & C. Hoyles & R. Noss & L. Edwards (Eds.), *Computers for Exploratory Learning*. Berlin: Springer–Verlag.

Schecker, H. P. (1993). The Didactic Potential of Computer Aided Modeling for Physics Education. In D. L. Ferguson (Ed.), *Advanced Educational Technologies for Mathematics and Science* (pp. 165-207). Berlin: Nato Conference, June 1990. Springer-Verlag.

White, B. Y. (1984). Designing Computer Games to Help Physics Students Understand Newton's Laws of Motion. *Cognition and Instruction, 1*(1), 69-108.

---

[i] The Playground Project, co-ordinated by the Institute of Education, University of London, is an EU ESPRIT project (No. 29329): http://www.ioe.ac.uk/playground (For a discussion of children's understanding of rules, see Hoyles, Noss, Adamson, & Lowe, submitted.)