Designing for learning mathematics through programming: a case study of pupils engaging with place value

Laura Benton*, l.benton@ucl.ac.uk, +44 20 7907 4627; Piers Saunders*, piers.saunders@ucl.ac.uk; Ivan Kalas*^, i.kalas@ucl.ac.uk; kalas@fmph.uniba.sk; Celia Hoyles*, c.hoyles@ucl.ac.uk; Richard Noss*, r.noss@ucl.ac.uk

* UCL Knowledge Lab, UCL Institute of Education, 23-29 Emerald Street, London, UK

^ Dept. of Informatics Education, Comenius University, Bratislava, Slovakia

Abstract

This paper focuses on a major part of a two-year intervention, ScratchMaths (SM), which seeks to exploit programming for the learning of mathematics. The SM hypothesis is that given the right design of curriculum, pedagogy and digital tools, pupils can engage with and express important mathematical ideas through computer programming. We describe the overall design of SM and as an illustration of the approach, we elaborate a more detailed description of the specific SM activities that seek to harness the programming concept of 'objects communicating with one another' for the exploration of the mathematical concept of *place value* through a syntonic approach to learning. We report a case study of how these activities were implemented in two primary classes. Our findings constitute a kind of existence theorem: that with carefully designed *and* sequenced learning activities and appropriate teacher support, this approach can allow pupils to engage with difficult mathematical ideas in new, meaningful and generalisable ways. We also point to the challenges which emerged through this process in ensuring pupils encounter these mathematical ideas.

Keywords

Children's programming; computing education; mathematics; Scratch; broadcasting; place value

1   Introduction

There was a body of research undertaken during the 1980s and 90s that explored the potential beneficial impact of learning to program on pupils' mathematical thinking [1]. Researchers designed mathematical activities that exploited the affordances of a specific programming environment to explore and express a range of mathematical ideas through the programming language. Many of these researchers were inspired by the work of Seymour Papert, and built microworlds in the Logo programming language [1]. Their research was guided by a constructionist approach to learning, which was viewed as a process of building knowledge structures that happens most effectively when the learner is actively engaged in "constructing a public entity" [2]. The approach endeavours to ensure that the learning activities embed "powerful ideas" [3] – ideas that

1

are potent in their use (both epistemologically and personally), in their connections with other disciplines and their fit with a learner's personal intuitive knowledge [4].

Researchers also identified challenges experienced in learning to program, for example Lewis [5] and Resnick et al. [6] pointed to limitations of the programming environment, such as attention on mastering the programming syntax rather than the semantic meaning of the code as well as the specific skills/knowledge required by teachers to provide the necessary guidance or challenge to novice learners. New blocks-based programming languages, such as Scratch (derived from Logo), help to overcome difficulties in syntax, thus making programming more accessible to a wider range of learners and teachers [7]. These languages are designed to make some complex concepts more accessible, with visual cues such as colour, shape and constrained nesting to indicate usage, flow and scope [5, 8].

The research reported here is based on the ScratchMaths (SM) project, which set out to exploit the resurgence of the teaching of programming (now commonly referred to as 'coding') in primary schools in England along with the programming functionalities for younger learners that have become available. Several curriculum designers have recognised the potential that Scratch offers the learning of mathematics, largely with a focus on geometry, but also exploring positive/negative numbers, building arithmetic and algebraic expressions, using the in-built coordinate system as well as developing mathematical thinking skills and positive attitudes towards mathematics [9-14]. SM aimed to make problematic parts of the mathematics curriculum personally meaningful, enabling more pupils to gain a deeper understanding of mathematical ideas through the programming activities used to express them. A central design focus was to exploit *syntonic learning opportunities*, that is to support learners in considering a problem from the perspective of the programmable object, which "represents a 'resonance' between external forms and concepts and what people know about themselves" [3, 15]. The concept of syntonicity has typically been used when exploring geometric ideas through programming [3]. We sought to extend its scope to other areas of mathematics and through a design research process aimed to identify concepts in the primary mathematics curriculum that we hypothesised would benefit from this approach.

The Scratch environment allows multiple objects (called sprites) to be programmed so that they act in parallel by programming interactions between them through a kind of message-passing, known as 'broadcasting'[1]. In this paper we explore how to exploit this programming functionality of objects communicating with one another as an alternative, meaningful and generalisable way for pupils to engage with the mathematical concept of place value - where the places 'interact', allowing pupils to play with the ideas directly rather than simply learning about them. We first present the background to the SM project and the overall design of the SM curriculum, before turning to the case study research that forms the empirical core of the paper.

2   Background

Learning something in one context, such as programming, and subsequently utilising this skill or knowledge in another context is commonly referred to as 'transfer' (see for example [16, 17]). The traditional view of transfer is that that learning x brings about an

---

[1] The key attributes of which are adopted from more general mechanism of event handling (in particular the mechanics of user-defined event handling)

ability to understand y where, in some real sense, x and y are epistemologically different, an approach that has been subjected to critical scrutiny (e.g. [18]).

The early Logo experiments constituted a widespread attempt to explore the impact of computer programming on children's mathematical understanding [1, 19]. Clements and Sarama [20] propose that programming in Logo could "serve as a transitional device between concrete experiences and abstract mathematics". However, there were conflicting findings, which may in part be explained by the different approaches to and reasons for teaching programming, as well as to the differing levels of facilitation of the connection-making process between the different domains, usually offered by a teacher [see 21, Ch. 7]. In fact Delclos et al. [22] state that "no content, standing alone, can spontaneously produce generalisable learning" and highlight the important role of the teacher in mediating the learning through programming languages and supporting transfer.

Some researchers have identified teaching techniques that facilitate transfer, such as *hugging* and *bridging*, distinguished by Salomon and Perkins [23]. Hugging seeks to make the teaching in the new domain as similar as possible to the original context [23-25]. Bridging by contrast promotes facilitating abstraction and connection-making processes by, for example, making analogies or teaching problem-solving strategies. Salomon and Perkins [23] suggest that using a combination of these techniques maximises potential transfer opportunities. Table 1 summarises a set of hugging and bridging techniques, based on the mechanisms for transfer proposed by Salomon and Perkins, and expanded by Fogarty et al. [26] . It lists 'ten tools for teaching for transfer' adapted specifically for mathematics by Jones et al. [27], and how these techniques have been used within programming and mathematics.

Hugging Techniques	Bridging Techniques
1. *Setting expectations:* indicating how to directly apply learning content in transfer context (e.g. identifying difficulties of fractions and discussing how they could be represented within Logo [28])	6. *Anticipating applications*: discussing potential transfer contexts of learning context (e.g. setting a task to program a mathematics game in Scratch to facilitate number understanding of younger pupils [29])
2. *Matching*: adjusting learning activity to reflect transfer context (e.g. identifying squares in environment and then using Logo programming to draw squares [3])	7. *Generalising concepts*: identifying generalisable aspects (e.g. principles, rules, ideas) of learning content (e.g. sequencing questions to encourage generalisation of algebraic concepts encountered in Logo [30])
3. *Simulating*: approximating learning content to transfer context through role play, acting out etc. (e.g. encouraging a pupil to "play Turtle" to imagine how to draw a square in Logo [3])	8. *Using analogies*: representing the learning content through a different topic using an analogy
4. *Modelling*: showing or demonstrating use of learning content within transfer context (e.g. providing examples of angles with rotation to support drawing of shapes within Logo microworld [31])	9. *Parallel problem-solving*: exploring parallels and differences in applying learning content within two different problem contexts
5. *Problem-based learning*: requiring the use of learning content to solve a problem within the transfer context (e.g. learning problem-solving skills through making	10. *Metacognitive reflection*: promoting planning, monitoring and on their own thinking (e.g. design debriefings to encourage reflection on connections

| games [32]) | between game design choices and maths learning [14]) |

Table 1 - Ten Tools for Teaching for Transfer [based on 26, 27] and examples of these techniques found within the existing research on teaching programming and mathematics

We do not see mathematical learning as a spin-off from learning to program, therefore we have considered whether children taught to program in Scratch can express some of their mathematical thinking in programming with the support of carefully designed materials *and* teachers who have been 'trained' in how to *use* Scratch to *learn* mathematics. Within mathematics there is often a disconnect between concept and procedure, with a dominant focus on the latter, despite overwhelming consensus that what is needed is a judicious mix of the two [33]. The difficulty for the pupils is to 'keep hold' of what the symbols *mean,* and simultaneously to let go of the meaning when manipulation of the symbols *is* a realistic objective[2]. A pervasive example of this is in the way children are introduced to the algorithms of arithmetic, which is the focus of our investigation here.

3   ScratchMaths Curriculum Design

In the SM project a two-year curriculum was designed for the 9-11 primary age group (Years 5 and 6), aligned to the English National Computing and Mathematics Primary Curriculums [34], and promotes the teaching of carefully selected core ideas of computer programming alongside specific mathematical concepts. The content was divided into six modules, three modules to be taught per year (see Table 2). The project team followed a design research process to develop the curriculum content and pedagogic approach (termed the 5Es framework[3]), trialling the activities in a number of 'design schools' and iteratively redesigning based on the outcomes of these trials (for further details see [35, 36]). The final version of the SM curriculum has recently undergone a randomised-control trial (RCT) which involved over 111 schools across England and undertaken by independent evaluators. The results of this evaluation are due to by published in Summer 2018 and are not within the scope of this paper.

---

[2] For example, a Logo program such as REPEAT 3 [FD 50 RT 120] contains within itself some information about the number of symmetries, and an application of a very powerful theorem (the Total Turtle Trip theorem). In short, the symbols bring alive aspects of the *structure* of an equilateral triangle.

[3] Which includes the constructs Explore, Explain, Exchange, Envisage and bridgE
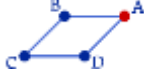
Year 5 Modules (ages 9-10)	Year 6 Modules (ages 10-11)
Module 1: Tiling Patterns	*Module 4: Building with Numbers*
Introduces the key computational concepts of sequencing, repetition, algorithm, debugging and definition as well as linking to symmetry, angles and negative numbers through building circular patterns.	Focuses on place value and requires the use of *broadcasting* to build place value models within several different contexts such as time and measurements.
Module 2: Beetle Geometry	*Module 5: Exploring Mathematical Relationships*
Focuses on creating different drawings using the pen tool such as regular polygons, introducing pupils to initialisation, expressions and randomness as well as consolidating earlier concepts.	Explores different types of mathematical relationships including proportionality and ratio as well as introducing the concept of variable.
Module 3: Interacting Sprites	*Module 6: Coordinates and Geometry*
Focuses on parallelism and building behaviours for multiple sprites, firstly isolated reactions to an event (when this sprite clicked), using conditions and expressions linked to coordinates, and then implementing interactions between multiple sprites using *broadcasting*	Centres around coordinates within multiple contexts firstly to investigate emerging shapes through the use of randomness, then to manipulate triangles and quadrilaterals and finally to explore different transformations such as translations and reflections.

Table 2 - Overview of SM Modules and Topics[4]

3.1 *Year 5 SM curriculum*

The first year of the curriculum (for Y5 pupils) primarily focused on developing pupils' programming skills, guided by the statutory National Computing Curriculum objectives [34] as well as the specific functionalities of Scratch. The links to mathematics were made explicit in an extensive set of teaching materials developed to support the delivery of the SM curriculum. Hugging and bridging techniques were suggested over both years, with the Y5 curriculum incorporating more *bridging* techniques to support pupils' in reflecting on the knowledge gained through programming and making conceptual

---

[4] The complete SM curriculum materials are available to download from the UCL website: http://www.ucl.ac.uk/scratchmaths

connections to their evolving mathematical knowledge. This is primarily operationalised through teacher-mediated bridging discussions to encourage generalisation and metacognitive reflection (for example in the case of 360° as a full turn).

3.2   Year 6 SM curriculum

All students in England sit a high-stakes national mathematics test at the end of Y6, so schools and teachers expect a clear focus on mathematical content during Y6. Thus the second year of the SM curriculum for Y6 pupils foregrounded the mathematics content with each module centred around the exploration through programming of a key and often challenging concept from the mathematics curriculum. Simultaneously the curriculum reinforced and built on the programming skills and knowledge developed during the previous year. The Y6 curriculum included more *hugging* techniques, where the programming activity goal clearly links to mathematics curriculum topics, by for example designing on and off-computer simulations of the SM models and matching some activities to typical mathematics test questions.

In this paper, we illustrate the design research that underpinned the SM activities exploring place value, a topic chosen as a notoriously problematic concept for pupils, and one which is tempting to concentrate *only* on procedural manipulation at the expense of the decimal structures that give rise to it. We aimed to exploit the functionality of being able to program interactions between objects in order to provide another way for pupils to engage with the mathematical concept of place value that is both meaningful and generalisable.

Below we briefly review research into pupils' understanding of place value and the rationale for exploiting the idea behind broadcasting.

3.3   Place Value

Place value is a key mathematical concept used to describe "the value assigned to a digit according to its position in a number" [37] and is fundamental to later mathematical attainment [38]. Understanding place value requires learners to integrate three abstract ideas: the quantities and base name (e.g. 1 hundred, 3 tens, 2 ones), the written numeral (e.g. 132), and the number name (e.g. "one hundred and thirty two")[39]. The reduction of place value to these three ideas, a way to read, write and say numbers, hides the underlying complexity of the concept [38, 40]. Place value is important for pupils as they develop an understanding of the decimal number system and are able to apply this understanding when performing more sophisticated multi-digit operations or make connections with novel problems [41, 42]. Researchers and practitioners have suggested that the concept of place value can be challenging for many children [37, 43], particularly when extending to other contexts such as money [44].

A crucial aspect of the concept is the need to grasp the *relationship* between the adjacent columns (e.g. ones/units and tens) and what happens to a multi-digit number ending in 9 when it is increased by 1 – the point at which this relationship becomes explicit [37, 45]. This process is described as carrying, exchanging or trading[5]. The conceptual abstraction required to construct the relationship between the groups of objects represented by each digit makes this process particularly challenging [43]. Fuson [45] also highlights the issue of an "exaggerated emphasis on instrumental knowledge at the expense of relational knowledge" in the teaching of place value. Within school-based

---

[5] In this paper we use the terms *carrying*

mathematics the process of carrying is often taught procedurally through an algorithm which children are expected to memorise and apply to multi-digit computation. However, researchers have highlighted that being required to follow a set method could hinder children's development of numerical reasoning and discourage logical thinking. Thus it is important to go through the process of constructing the algorithm for themselves at some level [46]. A comprehensive review of teaching and learning of whole numbers in primary schools documents the difficulties children continue to have in learning multi-digit addition and subtraction and, crucially, argues for the critical importance of 'teaching for meaning' in this area [47]. These difficulties, coupled with the fact that it is a mathematical topic that is revisited in every year of primary school in England, led us to select place value as a mathematical focus for the first Y6 module (Module 4).

In SM, a key learning activity design principle was exactly to teach for meaning, and that the chosen mathematical concepts should be "experienced through purposeful activity that engenders a sense of the power or utility of the concept" [47, 48]. Given our emphasis on syntonicity we hypothesised that being able to "act out" the process would enable pupils to better identify with the algorithm. We therefore decided that objects should be used to represent the place value digits and these digits could then be programmed to model the carry behaviours themselves using Scratch's broadcasting behaviour. Through further development of the learning activity design we subsequently began to uncover the power of this idea to extend to decimals as well as beyond the base 10 model to other contexts such as time through building stopwatches (and countdown timers).

3.4   Building a Place Value Model

In Scratch objects (known as sprites) can be programmed to act in certain ways and to trigger actions of other objects. Objects can communicate with each other by 'broadcasting' a 'message' to every object within the Scratch environment, and the chosen recipient(s) configured to receive the message and then act upon it. These processes are implemented by a pair of blocks: sending a message by the **broadcast message** block[6] and reacting to it (by one or several sprites) by the **when I receive message** block.

Figure 1 - Blocks used for broadcasting in Scratch

In the SM curriculum, the concept of interacting sprites through broadcasting was introduced in the previous year (Y5) through an unplugged (off-computer) activity, using the metaphor of a poem, with the different lines of the poem acting as the message to be sent (i.e., read aloud), listened for and received (resulting in the next line being read). Pupils then move onto computer-based activities where they build interactions between sprites, firstly with only one sprite reacting to a broadcast and then multiple sprites reacting to the same broadcast. Thus from a computational point of view, pupils are exposed to the key programming ideas of parallelism, events, multiple sprites and interactions between them.

In Module 4, Y6 pupils return to broadcasting to build a multi-digit Scratch counting model, which they later extend to the context of time in building a stopwatch (see Figure

---

[6] Note that the broadcasting block does not specify an 'addressee'

2 for overview) initially using **ones** and **tens** sprites to represent place value digits[7] (and subsequently **1 secs** and **10 secs** sprites). *Expectations are set[8]* by making direct connections with the place value model, with pupils *simulating* the carry process by acting as digits and "nudging" the person standing next to them who has been 'programmed' to receive the carry each time they reached the 0 digit of their flip book (Investigation 1 Activity 2 within Figure 2 and top image of Figure 3). Each sprite has ten digits, with one digit displayed as its costume (see bottom image of Figure 3). Scratch costumes (i.e. different appearances of the sprite) can be iterated in order, using the **next costume** block or changed to display specific costume using the **switch costume to...** block.

Figure 2 - Overview of core Module 4 activities (activities classified as 'extensions' are not included)

---

[7] Note in the mathematics curriculum the first column of a whole number displayed in base 10 is called 'ones', not 'units' as sometimes designated and the second 'tens' etc.
[8] Specific hugging techniques are highlighted in *italics*

Figure 3 – Flip books used in Investigation 2 Activity 2 (top) and the ten costumes of each sprite (bottom)

A Scratch program (known as a 'script') needs to be written to simulate adding, so that when the **ones** sprite is clicked it goes to the next costume, which increases it by one. The activity encourages the *matching* of the clicking 'action' to addition (and later subtraction). When the tenth costume is reached (which appears as a 0), the **ones** sprite if programmed correctly should broadcast a message "add 10". The **tens** sprite is programmed to receive this specific message at which point it increases its 'value' (i.e., its costume) by one. This models the carry process typically used in schools. In SM the last step is given as a pupil challenge: that is to write the program first for the **ones** sprite, including to work out at which point the carry needs to occur, and second for the **tens** sprite[9]. These two models can then be extended by the pupils to other place values (up to 1000[10]) as well as decimals and *generalised* to perform conversions in time and measure (see Investigations 2-4 in Figure 4)[11].

---

[9] Although we provide guidance for the positioning of the sprites via a background template, the sprites can be positioned anywhere on the screen whilst still representing their original place (identified by their name) and increased when the **ones** sprite is clicked. As the connections are made through the broadcasting process and the sprite acts as a representation of the digit (place) then the pupils' are challenged in their understanding to go beyond the basic carry algorithm and consider how they are constructing their representation of the digit value

[10] This model is generalised by changing the message that is broadcast/received so the **hundreds** sprite is listening for "add 100" and the **thousands** sprite is listening for "add 1000" in this way pupils specify the specific digit that will be affected by the carry process

[11] Note that the model is built so that the total can only increase when the **ones** sprite is clicked.
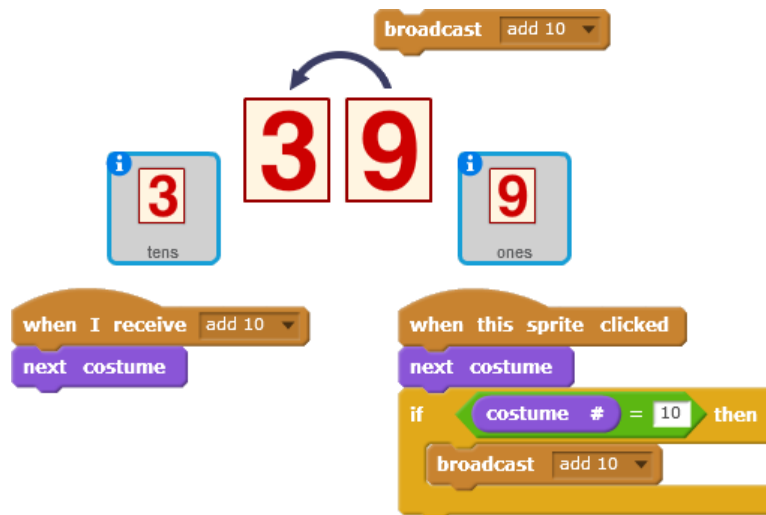
Figure 4 – Scratch program demonstrating the use of broadcasting as carry in the place value model. The ones sprite increases by one (i.e. shows the next costume) every time it is clicked and when it reaches the tenth costume (which is 0) it broadcasts a message which is received and acting upon by the tens sprite making it increase by one.

4   Methodology

Our hypothesis was that it would be possible to use the mathematical algorithm for addition expressed through programming as a means to develop understanding of the place-value concept. Specifically, our research question was: Do pupils meaningfully engage with the mathematical concept of place value both procedurally and conceptually by exploiting the programming functionality of objects that are able to communicate with one another, and if so how and what is the impact? We chose to follow a single qualitative case-study approach, to establish the feasibility of our hypothesis and investigate in-depth how pupils engaged with and expressed the mathematical ideas within place value.

4.1   Participants

The school selected for our case-study was a large two-form entry[12] primary in London that was participating in the RCT. It was chosen as an example of best practice in terms of engagement with SM – the class teachers had attended all of the required professional development and delivered the core content from each of the modules to date.

The case-study involved two Y5 teachers (Teachers A and B) and two Y6 teachers (Teachers C and D). The Y5 teachers delivered the first three SM modules and despite limited previous experience with Scratch reported being confident in delivering the SM curriculum to their pupils. However, as these pupils moved into Y6 two *different* Y6 teachers from the same school took over the SM teaching and delivered the Y6 modules. The Y6 teachers were less confident with technology and initially found the technical side of the teaching more challenging than the Y5 teachers, but were supported by the Y5 teachers with whom they co-taught the initial lessons.

---

[12] two classes per year group

4.2 Data Collection

A researcher visited the school during Module 3 and observed one lesson (for each Y5 class) which introduced to the concept of broadcasting. The same researcher returned to the school six months later to observe the same two classes of pupils (now in Y6) being taught a lesson from Module 4 (Investigation 2 Activity 1 in Figure 2) which involved using broadcasting to build a place value model of time in the context of building a stopwatch.

The researcher audio-recorded what happened in each lesson and took detailed notes to supplement the recording as well as collected all of the pupil work (Scratch projects). The researcher probed with the pupils during the lessons to ask them to explain *how* they were implementing their Scratch projects. The audio recordings from the lessons were transcribed and collated with the field notes.

4.3 Analysis

The analysis was split into two parts:

    (i)   establishing the extent to which pupils could successfully implement the broadcasting functionality (in Y5) and whether this learning was subsequently exploited in the specified mathematical context (in Y6)

    (ii)   identifying the different hugging and bridging techniques that the teachers employed during the Y6 lesson and the potential impact on pupils' engagement and expression of place value.

To address the first part, a content analysis [49] of the pupils' Scratch projects was undertaken (each project represented the output of a lesson – approximately 1 hour). The analysis focused on the success of the implementation of the different aspects of the broadcasting functionality to support (a) interactions between sprites and (b) a working place value model (Y6 only) as well as identifying any reasons for not succeeding in this.

Next, to identify the hugging and bridging techniques the transcripts and field notes from the lesson observations were imported into nVivo and following a thematic analysis approach, deductively coded using a coding framework based on Table 1. During the first review of the data set a researcher identified and coded all potential examples of hugging and bridging. The resulting data subset was then reviewed for a second time by the same researcher, who identified and coded any instance of a specific hugging and bridging technique (as set out in the coding framework - see Table 5 for an overview of the output of this coding process).

5 Results

5.1 Broadcasting – implementation and exploitation for mathematics

Both Y5 teachers followed the SM lesson plan closely, undertaking the suggested activities to introduce the concept of broadcasting as described earlier and using the SM classroom presentations to guide discussions. During their lessons both teachers specifically focused on the following key attributes of broadcasting:

- *Sending a message*

- *Listening* for a particular message
- *Receiving* that message and reacting in some way

The content analysis of the Y5 pupils' projects revealed that the majority of pupils in both classes were able to construct successfully a functioning broadcast event (see Table 3), i.e. one sprite receiving a broadcast message from another sprite and reacting, and/or multiple sprites receiving the same broadcast message and reacting in parallel. Only three pupils (out of 49) did not achieve these objectives: one used broadcast blocks but did not use any blocks to receive the message; the other two did not incorporate the **broadcast** blocks into their Scratch programs and also set their **when I receive…** blocks to listen for a message that was never broadcast.

	No. pupils	Total projects	% Success rate (no. of pupils)	One sprite reacts	Several sprites react	Did not achieve
Class 1 Teacher A	23	13	85% (11)	9 projects	5 projects	2 projects
Class 2 Teacher B	26	12	100% (12)	9 projects	9 projects	0 projects

Table 3 - Number of pupils building broadcasting behaviours that are reacted to by one sprite or multiple sprites (pupils generally worked in pairs, but a few worked individually or as a three)

During the second observed lesson (Y6) Teacher D followed the SM classroom presentation slides, but Teacher C provided much less structure for her class - she initially discussed what a stopwatch should do but then left the class to explore for themselves without any visual reinforcement or guidance from the slides. In both classes, pupils helped one another, which resulted in similar solutions.

Table 4 shows that more pupils in Class 2 were able to build a working stopwatch than in Class 1 (i.e., to be able to broadcast between two adjacent sprites to count above nine). One of the most common reasons for the stopwatch not working was an incorrect positioning of the broadcast block often causing the **tens** sprite to increase too late e.g. resulting in the sequence 09, 00, 11. Some pupils were also missing the corresponding receive block in the **tens** sprite or had too many broadcast blocks. A few pupils had issues with their stopwatch, which were unrelated to the broadcasting functionality such as putting their programs within a repeat 0 block (which would not run) or referring to digit costume numbers within a program that had been deleted.

	No. of pupils	Total projects	% Success rate (working stopwatch)	Incorrect broadcast block position	No receive block	Too many broad-casts	Other issue - repeat 0 - deleted costumes
Class 1 Teacher C	23	11	27% (3)	3 projects	2 projects	1 project	2 projects
Class 2 Teacher D	27	14	64% (9)	3 projects	2 projects	0 projects	0 projects

Table 4 – Number of pupils who successfully used broadcasting to build a stopwatch and reasons for not succeeding (note – one pair from Class 1 was excluded from this as they accidentally used the teacher's final project and so did not reflect their own work)

Some pupils independently followed an alternative approach to building their stopwatch, using a series of **switch costume to…** and **wait** blocks with a broadcast at the end of each sequence of 10 switches that simulates the synchronised interactions of sprites but limits opportunities for both generalisations. They had a block representing

every single step of the process, resulting in long complex programs. However, it does work and one boy at least saw it as an efficient way to implement his stopwatch "*because you can see how it's going to work*". Another pupil used a single broadcast when initialising his stopwatch, which then set off a series of **wait** blocks in each of the sprites (see Figure 5). The problem with both of these approaches is that although the stopwatch visually worked (or in the Figure 5 solution would work if the left program waited for 10 secs) there was no *link between the different digits,* the interactions were concealed which meant that this pupil finessed the concept of the carry process. This demonstrates a failure to focus on the structure of the model.
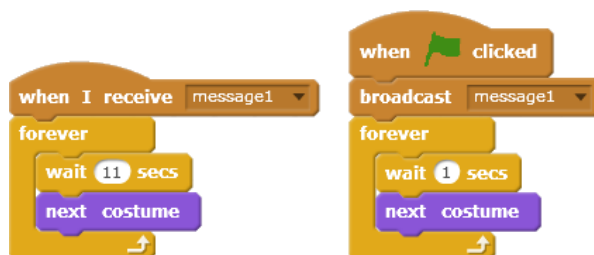

Figure 5 - Approach using wait to control stopwatch: Scratch programs for tens sprite (left) and for ones sprite (right)

5.2 Hugging and bridging techniques

The two Y6 teachers followed the same lesson plan, but applied their own pedagogical strategies to lesson delivery (which drew from different areas of the SM 5Es framework), and included different hugging and bridging techniques (see Table 5). Teacher C employed an open *problem-based learning* technique throughout, where she set her pupils the task of building a stopwatch without providing additional structure or guidance, instead she prompted them to explore and extend their knowledge from the earlier activity of building a place value model. In contrast, Teacher D followed through the SM classroom presentation slides, she read out the activity instructions directly from the slide and asked the suggested questions. However, both teachers drew from the 5Es pedagogical framework, for example they provided pupils opportunities to 'exchange' their work with classmates and 'explain' the challenges they had experienced.

The teachers also both used a *matching* technique multiple times throughout their lessons, they referred to the digit sprites as units, tens and hundreds as well as termed their position 'columns' and made links between the numbers on a stopwatch and the costumes in Scratch. In both classes this resulted in a discussion around which costumes were needed for the **ten secs** sprite and whether would the stopwatch ever display 60 seconds, which highlighted the alternative representation of 60 seconds as 1 minute on a stopwatch.

There were instances of the teachers giving an 'active voice' to the sprites. Teacher D talked about the carry process as the sprites 'nudging' each other, she reflected the vocabulary used within the earlier unplugged activity, and Teacher C used an actor *analogy* to explain the process of addition and carrying, with a costume change representing each increment and communication between two actors representing the carry process. Many of the pupils talked about the carry process in a similar way; for instance one pupil described it as the digits having a conversation where "*number 9 tells number 10 oi* (sp.) *go on move*". Other pupils viewed it more as telling someone to do something: "*the message goes somewhere else to that person and to that tens column*" and

a further pupil explained "*it's telling the next column that that's when they have to start moving…make it into a 2 digit number*".

Hugging Techniques	Teacher C Example(s)	Teacher D Example(s)
Matching	"*What are these digits called? They are in the units column, but when you're thinking about ScratchMaths what do we call them?*" "*If this is a stopwatch what costumes did you need?*"	"*If you think of your place value in the units column what are we hoping to see?*" "*That's it because 60 seconds in a minute and it should automatically nudge so that that increases*"
Modelling	Not used	[Points to a stopwatch image displaying 00:00 on the interactive whiteboard] "*If this was 00:09 what should have happened?*"
Problem-based learning	"*I would like you to see if you can figure out using the knowledge from digits up…can you work out how to make a stopwatch. So the numbers increase in the same way, but you're thinking about the time*"	Not used
Bridging Techniques		
Generalising concepts	Not used	[In response to a pupil's project who had not simplified his program] "*so you've actually written out everything step by step by step rather than say multiplying 24 x 2, you've written 2 x 2, 3 x 2, 4 x 2*"
Using analogies	"*so this is the first actor on stage and he's changing his costume continuously in ones all the way up to 9*" "*…when he's finished changing his costumes he calls on the next actor and they change*"	Not used

Table 5 – Examples of the hugging and bridging techniques used by the two Y6 teachers

6 Discussion

Here we return to our original research question and consider pupils' engagement with the concept of place value as well as the impact of this and the challenges in fostering meaningful engagement. Our findings show the majority of pupils were able to engage with and implement a challenging programming concept – the communication of multiple objects (i.e. sprites) with one another, and that many of these pupils were able to build on this concept and use it to explore in a mathematical context, giving syntonic expression to what is generally a procedural process of carrying.

We note a striking facet of pupil and teacher responses to our intervention: namely the extent to which some pupils managed to *place themselves into the action*, giving life to the mathematical relationships by constructing narratives for the behaviour of the digits. We discern a pattern here that is a type of syntonicity, precisely the kind raised by Papert [3]. What we have in SM is a kind of generalisation of the syntonic idea. When a pupil says "*it's telling the next column that that's when they have to start moving…make it into a 2 digit number*", she is describing what the numbers are saying to each other, why they 'have to start moving' and when they need to do so. She is putting herself into the action, or allowing herself to consider and express the action coming to her. This stands in stark contrast to what is made feasible with the inert technology of pencil and paper, where the numbers have rules they have simply to obey.

We also saw how the teachers were *using* the shared language of the Scratch context to explain and elaborate, using various hugging techniques, how the system works, how and why place-value is central, and how the numbers are programmed (using the costume metaphor) to instantiate it. In addition, the shared experience of acting out the process helped teachers explain the algorithmic process. In subsequent work with other SM schools teachers have expressed how this process of hugging provides opportunities for the identification of issues with pupil understanding of place value that would not be identified within conventional mathematics: "*Some of them have no idea why they are moving the numbers across it's just because they've been told to do it…they've been doing it right but they can't necessarily understand why and it picks out those children because if they haven't got that…* [For example] y*ou are at 12.9 and then they've got 12.10 you can explain that through the SM … if they have got that understanding of what happens when you get to 9 and then you add 1 and it moves up, then they can explain that through Scratch.*"

Various existing initiatives have helped children grasp the idea of place value but the difficulty remains in supporting pupils to move beyond procedural knowledge [47] as well as generalise to other areas such as time and measure, which our findings demonstrate a first step towards.

Our findings also highlight several challenges. The analysis of the Y6 pupil projects exhibits varying success in building the place value model in another context, namely time. A greater number of pupils in Class 2 were able to build their model; their teacher provided more structure and followed the steps of activity design more closely using the provided resources and embedded hugging techniques which suggests that the increased structure and guidance may have contributed to the success of these pupils.

We looked in detail at the specific issues experienced by unsuccessful pupils. One of the principles of Scratch is for all blocks to be, what Maloney et al. [50] term *failsoft*, which means rather than displaying error messages it attempts to "do something sensible" with that block. Therefore attention is not drawn to syntactic issues experienced by pupils, such as referencing a sprite costume number which did not exist due to being deleted. Furthermore, pupils experiencing problems such as broadcasting a message with no receive block or having blocks that are not run (due to repeat 0) would not receive any feedback to help them identify these issues. This highlights the importance of the curriculum identifying these issues so teacher and peer support can be harnessed to explain them.

Additionally, there were pupils who followed approaches, which although achieving a working stopwatch could be viewed as mathematical dead-ends. Gurtner [25] proposed (with reference to Logo) an extension to the Salomon and Perkins notion of transfer, adding a further metaphor of 'tunnels'. He suggests that, certain features of the

programming environment might enable pupils to make progress through mathematically challenging areas but also hide interesting connections with particular mathematical ideas. This was the case of the pupil who chose to use a series of **wait** blocks to build his stopwatch and also to some extent the pupil who was not able to simplify his complicated program, obscuring the mathematical idea of generalisation.

These challenges reinforce previous claims that simply being exposed to the programming environment does not automatically result in connections with the mathematical ideas [1, 22]. Having a combination of carefully designed activities and teacher guidance, which provides explicit hugging and bridging techniques, is crucial along with teachers who understand the power of these techniques.

7 A concluding remark

Our experiences in developing the SM intervention demonstrate the significant task of matching the learning content and progression in two curriculum areas rather than taking a more patchwork approach. It thus highlighted the necessity of following an iterative design research process to ensure fit with pupil developmental levels and accessibility for all. Furthermore, programming concepts need to be carefully selected and introduced so that they can later be extended to key areas of mathematics, becoming powerful ideas instead of mathematical dead ends. As well as place value and broadcasting, the SM project has designed activities to exploit programming with *variable* to explore ratio and proportion as well as *pick random*[13] to explore coordinates.

Our research has demonstrated that this approach is not without its challenges, not least what Noss and Hoyles [21] term the 'play paradox' - the problem of designing a learning activity to allow pupils to explore and construct ideas for themselves, but also ensure they encounter the important mathematical ideas embedded within the programming activities. It is clear the teacher plays a crucial role in highlighting and supporting these ideas as well as in providing on-going feedback. Further research needs to be undertaken in developing more specific effective teacher hugging and bridging strategies to facilitate pupil engagement with and expression of further mathematical concepts.

This work has shown the potential of the SM approach for a wide range of pupils to engage with mathematics through both a new medium, i.e. a computer-programming environment instead of paper and pencil, as well as from a new perspective through a syntonic approach to learning. We have observed high levels of engagement and enthusiasm from pupils in their SM lessons. As one pupil explained "*What we really like about Scratch is when you press that start button and you see your script come into life, it's like magic in front of your eyes*". We now have an opportunity to exploit this potential for the benefit of mathematics as well as to further explore how to extend these ideas to other areas of the curriculum – not just learning to program for programming's sake, but to ensure this time around programming truly becomes a purposeful medium of expression within our education systems.

---

[13] an operator in Scratch which outputs a random value within a specified range

Acknowledgments

8   References

[1] Hoyles, C. and Noss, R. *Learning Mathematics and Logo*. MIT Press, Cambridge, MA, USA, 1992.

[2] Papert, S. and Harel, I. Situating constructionism. *Constructionism*, 36 (1991), 1-11.

[3] Papert, S. Mindstorms: Children, Computers, and Powerful Ideas (1980).

[4] Papert, S. What's the big idea? Toward a pedagogy of idea power. *IBM Systems Journal*, 39 (2000), 720-729.

[5] Lewis, C. M. *How programming environment shapes perception, learning and goals: logo vs. scratch*. ACM, City, 2010.

[6] Resnick, M., Maloney, J., Monroy-Hernández, A. R., N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B. and Kafai, Y. Scratch: Programming for all. *Communications of the ACM*, 52 (2009), 60-67.

[7] Weintrop, D. and Wilensky, U. *To block or not to block, that is the question: students' perceptions of blocks-based programming*. ACM, City, 2015.

[8] Weintrop, D. *Minding the gap between blocks-based and text-based programming*. ACM, City, 2015.

[9] Bagge, P. *How to Teach Primary Programming Using Scratch*. The University of Buckingham Press, 2015.

[10] Förster, K.-T. *Programming in Scratch and Mathematics: Augmenting Your Geometry Curriculum, Today!* ACM, City, 2015.

[11] Gadanidis, G. *Coding for Young Mathematicians, Grades 1-8*. Western University, 2015.

[12] NCTE/LERO *Scratch Programming and Numeracy in Senior Primary Classes*. City, 2012.

[13] Lewis, C. M. and Shah, N. Building upon and enriching grade four mathematics standards with programming curriculum. In *Proceedings of the Proceedings of the 43rd ACM technical symposium on Computer Science Education* (Raleigh, North Carolina, USA, 2012). ACM, [insert City of Publication],[insert 2012 of Publication].

[14] Ke, F. An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education*, 73 (2014), 26-39.

[15] Watt, S. *Syntonicity and the psychology of programming*. City, 1998.

[16] Perkins, D. N. and Salomon, G. Teaching for transfer. *Educational leadership*, 46, 1 (1988), 22-32.

[17] Pea, R. and Kurland, M. On the Cognitive Effects of Learning Computer Programming. *New Ideas in Psychology*, 2 (1983), 137-168.

[18] Lobato, J. Alternative perspectives on the transfer of learning: History, issues, and challenges for future research. *The Journal of the Learning Sciences*, 15, 4 (2006), 431-449.

[19] Clements, D. H. Computers in early childhood mathematics. *Contemporary issues in early childhood*, 3, 2 (2002), 160-181.

[20] Clements, D. H. and Sarama, J. Research on Logo: A Decade of Progress. *Computers in the Schools*, 14 (1997), 9-46.

[21] Noss, R. and Hoyles, C. Windows on mathematical meanings: Learning cultures and computers (1996).

[22] Delclos, V. R., Littlefield, J. and Bransford, J. D. Teaching thinking through logo: The importance of method. *Roeper Review*, 7, 3 (1985/02/01 1985), 153-156.

[23] Salomon, G. and Perkins, D. N. Rocky roads to transfer: Rethinking mechanism of a neglected phenomenon. *Educational psychologist*, 24 (1989), 113-142.

[24] Franklin, D., Hill, C., Dwyer, H. A., Hansen, A. K., Iveland, A. and Harlow, D. B. *Initialization in Scratch: Seeking Knowledge Transfer*. ACM, City, 2016.

[25] Gurtner, J.-L. *Between Logo and mathematics: a road of tunnels and bridges*. MIT Press, City, 1992.

[26] Fogarty, R., Perkins, D. and Barell, J. *How to Teach for Transfer: The Mindful School*. IRI/Skylight Training and Publishing, 1992.

[27] Jones, J. L., Jones, K. A. and Vermette, P. J. Teaching mathematics understandings for transfer. *Teaching Mathematics and its applications*, 28, 3 (2009), 145-149.

[28] Harel, I. and Papert, S. Software design as a learning environment. *Interactive Learning Environments*, 1 (1990), 1-32.

[29] Calder, N. Using Scratch: An Integrated Problem-Solving Approach to Mathematical Thinking. *Australian Primary Mathematics Classroom*, 15, 4 (2010), 9-14.

[30] Noss, R. Constructing a conceptual framework for elementary algebra through Logo programming. *Educational Studies in Mathematics*, 17, 4 (1986), 335-357.

[31] Cope, P. and Simmons, M. Children's exploration of rotation and angle in limited Logo microworlds. *Computers & Education*, 16, 2 (1991), 133-141.

[32] Akcaoglu, M. Learning problem-solving through making games at the game design and learning summer program. *Educational Technology Research and Development*, 62, 5 (2014), 583-600.

[33] Council, N. R. and Committee, M. L. S. *Adding it up: Helping children learn mathematics*. National Academies Press, 2001.

[34] DfE Computing Programmes of Study: Key Stages 1 and 2. *National curriculum in England* (2013).

[35] Benton, L., Hoyles, C., Noss, R. and Kalas, I. Building mathematical knowledge with programming: insights from the ScratchMaths project. In *Proceedings of the Constructionism in action: Constructionism 2016* (2016). Suksapattana Foundation, [insert City of Publication],[insert 2016 of Publication].

[36] Benton, L., Hoyles, C., Kalas, I. and Noss, R. Bridging Primary Programming and Mathematics: Some Findings of Design Research in England. *Digital Experiences in Mathematics Education* (2017), 1-24.

[37] Thompson, I. Teaching Place Value in the UK: time for a reappraisal? *Educational Review*, 52, 3 (2000), 291-298.

[38] Hurst, C. and Hurrell, D. Developing the big ideas of number. *International Journal of Educational Studies in Mathematics*, 1, 2 (2014).

[39] Payne, J. and Huinker, D. Early number and numeration. *Research ideas for the classroom: Early childhood mathematics* (1993), 43-71.

[40] Major, K. The Development of an Assessment Tool: Student Knowledge of the Concept of Place Value. *Mathematics Education Research Group of Australasia* (2012).

[41] Doabler, C. T. and Fien, H. Explicit mathematics instruction what teachers can do for teaching students with mathematics difficulties. *Intervention in School and Clinic*, 48, 5 (2013), 276-285.

[42] Verschaffel, L. and De Corte, E. *Number and arithmetic*. Springer, City, 1996.

[43] Kamii, C. Place value: An explanation of its difficulty and educational implications for the primary grades. *Journal of Research in Childhood Education*, 1, 2 (1986), 75-86.

[44] Chandler, C. C. and Kamii, C. Giving Change When Payment Is Made with a Dime: The Difficulty of Tens and Ones. *Journal for Research in Mathematics Education*, 40, 2 (2009), 97-118.

[45] Fuson, K. C. Issues in place-value and multidigit addition and subtraction learning and teaching. *Journal for research in mathematics education*, 21, 4 (1990), 273-280.

[46] Kamii, C. and Dominick, A. The harmful effects of algorithms in grades 1-4. *The teaching and learning of algorithms in school mathematics*, 19 (1998), 130-140.

[47] Nunes, T., Dorneles, B. V., Lin, P.-J. and Rathgeb-Schnierer, E. *Teaching and Learning About Whole Numbers in Primary School*. Springer, City, 2016.

[48] Pratt, D. *Making mathematics phenomenal: Based on an Inaugural Professorial Lecture delivered at the Institute of Education, University of London, on 14 March 2012*. Institute of Education, University of London, 2012.

[49] Stemler, S. An overview of content analysis. *Practical assessment, research & evaluation*, 7, 17 (2001), 137-146.

[50] Maloney, J., Resnick, M., Rusk, N., Silverman, B. and Eastmond, E. The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, 10, 4 (2010), 16.